CHAPTER

MULTIAGENT SYSTEMS: LEARNING, STRATEGIC BEHAVIOR, COOPERATION, AND NETWORK FORMATION

26

Cem Tekin[*], **Simpson Zhang**[†], **Jie Xu**[‡], **Mihaela van der Schaar**[§]

Electrical and Electronics Engineering Department, Bilkent University, Ankara, Turkey[*] *Economics Department, University of California, Los Angeles, Los Angeles, CA, United States*[†] *Electrical and Computer Engineering Department, University of Miami, Coral Gables, FL, United States*[‡] *Electrical Engineering Department, University of California, Los Angeles, Los Angeles, CA, United States*[§]

26.1 LEARNING IN MULTIAGENT SYSTEMS: EXAMPLES AND CHALLENGES

26.1.1 CONTENT AGGREGATION

Proliferation of web-based multimedia content sources and servers led to a tremendous growth in the volume and diversity of multimedia content that is consumed by a diverse set of users. This diversity results in users exhibiting a vast range of preferences over the content, which often depends on the context in which they consume the content. Such demand led to the emergence of *multimedia content aggregators* (MCAs) [1,2] that gather and fuse content from numerous multimedia sources to provide a ubiquitous content delivery experience for their users. It is thus essential for these systems to learn the context-specific content preferences of their users using past feedback of their users on the content that they provide. The context of a user includes information that is related to its content preferences, including but not limited to the location information, search query, gender, age, and the type of the device that the user is using (e.g., mobile phone, tablet, PC) to access the content [3]. Thus, the goal of the MCA is to match its users with the most appropriate content by learning how users with different contexts react to different contents. Such learning is necessary for continuous satisfaction of a user's request for content, which dynamically evolves over time depending on how the user's context evolves. This problem can be formulated as an online learning problem where the MCA learns the best content for its users through exploring how its users react to different content.

In order to maximize the user satisfaction, an MCA needs to connect with other MCAs that have access to other multimedia sources to find the right content for its users. This requires cooperation between the MCAs: In addition to serving its own users, an MCA should also serve content to the users of other MCAs when a request is made. Thus, each MCA has two types of users: direct users, who visit the website of the MCA to search for content, and indirect users, who are users of another MCA that requests content from the MCA. The objective of an MCA is to maximize the number of positive feedbacks (e.g., likes) given by its users (both direct and indirect). This objective can be achieved simultaneously by all MCAs through a distributed learning method, where all MCAs learn online how to match their current users with the right content (either by serving content from a directly connected multimedia source or by requesting content from another MCA) [4]. Learning the right matching using the past information collected from the users is not a trivial task because both the user and content characteristics are dynamically changing over time, and the most suitable content for the user might be located in the network of another MCA.

26.1.2 DISCOVERING EXPERTISE

An important multiagent learning problem is to assign arriving tasks to experts in a way that maximizes the performance (e.g., quality, accuracy, timeliness) of task execution. In this problem, each agent experiences inflows of tasks that needs to be completed. To achieve this, an agent needs to assign its tasks to appropriate experts. However, the most suitable expert for a task might be located out of the agent's own expert network. In such a case, the agent needs to request help from the other agents to match its task with the most suitable expert. In addition, the performance of an expert for a particular task depends on the context of the task, and how well an expert executes a particular task is not known in advance. Hence, the expertise of each expert in the system needs to be learned in a context-driven and decentralized way.

An interesting example of the multiagent learning problem mentioned above is the medical expertise discovery problem for diagnosis of patients with complex diseases. Developing new learning methods turns out to be essential for this problem due to the fact that the standard clinical methods often fail to provide an accurate diagnosis for these complex cases [5]. Fortunately, the widespread adoption of *electronic health records* (EHRs) allows for the aggregation of patients' records for use by an automated *clinical decision support system* (CDSS) that supports matching patients with the right medical expert. The decision-making process to match the patient with an expert of appropriate expertise should be guided by the available information about the patient, which may include the patient's past health condition, past hospital visits, drug administrations, and genetic and social data. This vast amount of information, also called the context, can greatly affect the diagnosis accuracy of the experts. For instance, it is possible that the experts residing in the same institution have experience confined to a specific population of patients that is admitted to the institution. In such a case, these experts will be unable to form an accurate diagnosis for a patient that has a context that is very different than the contexts observed from the other patients treated within the same institution. Therefore, it is essential to discover the level of expertise (diagnosis accuracy) of the experts for each patient context in order to match the patient with the right expert. Moreover, in order to further improve the diagnosis accuracy, the patient's institution may seek help from other institutions that might be better at handling this particular patient context. For instance, a small clinic may only have a primary care physician and may not have any specialists that are able to identify a particularly complex disease. Such a clinic may be able to request information from an established hospital that has numerous specialists and CDSSs that can be helpful in diagnosing the patient. In such a case, the control of the small clinic over the established hospital is limited. While the clinic can request information, it cannot select the expert that the hospital will assign to the case. Therefore, a learning mechanism that enables these institutions to cooperate by preserving their autonomy and privacy is necessary.

Numerous challenges need to be addressed in designing such a learning mechanism: Each agent should learn (i) whether it should rely on its own experts or request another agent to execute the task, and (ii) if it relies on its own experts, which expert it should assign to maximize the performance of the task execution. For instance, for the medical expertise discovery problem, the performance can be defined as a function of the diagnosis accuracy and the cost of diagnosis (time, money, etc.). In addition, learning should be continuous, i.e., the expert selection strategy needs to be updated every time after the performance of the task execution is observed to get the most benefit from the newly acquired information. Moreover, the contextual specializations of the agents and experts should be learned from past data. Because the number of different contexts is vast, intelligent mechanisms that aggregate information from similar past contexts should be developed.

26.1.3 POPULARITY FORECASTING

Social media has recently been used to provide situational awareness and inform predictions and decisions in a variety of application domains, ranging from live or on-demand event broadcasting to security and surveillance [6], health communication [7], disaster management [8], and economic forecasting [9]. In all these applications, forecasting the popularity of the content shared in a social network is vital due to a variety of reasons. For network and cloud service providers, accurate forecasting facilitates prompt and adequate reservation of computation, storage, and bandwidth resources [10], thereby ensuring smooth and robust content delivery at low cost. For advertisers, accurate and timely popularity predictions provide a good revenue indicator, thereby enabling targeted ads to be composed for specific videos and viewer demographics. For content producers and contributors, attracting a high number of views is paramount for attracting potential revenue through micropayment mechanisms.

While popularity prediction is a long-lasting research topic [11–14], understanding how social networks affect the popularity of the media content, then using this understanding to make better forecasts poses significant new challenges. Conventional prediction tools have mostly relied on the history of the past view counts, which worked well when the popularity solely depended on the inherent attractiveness of the content and the recipients were generally passive. In contrast, social media users are proactive in terms of the content they watch and are heavily influenced by their social media interactions; for instance, the recipient of a certain media content may further forward it or not, depending on not only its attractiveness but also the situational and contextual conditions in which this content was generated and propagated through social media [15]. For example, the latest measurement on Twitter's Vine, a highly popular short mobile video sharing service, has suggested that the popularity of a short video indeed depends less on the content itself, but more on the contributor's position in the social network [16]. Hence, being situation-aware, e.g., considering the content initiator's information and the friendship network of the sharers, can clearly improve the accuracy of the popularity forecasts. However, critical new questions need to be answered: which situational information extracted from social media should be used, how to deal with dynamically changing and evolving situational information, and how to use this information efficiently to improve the forecasts?

As social media becomes increasingly more ubiquitous and influential, the video propagation patterns and users' sharing behavior dynamically change and evolve as well. Offline prediction tools [11,17–19] depend on specific training datasets, which may be biased or outdated, and hence may not accurately capture the real-world propagation patterns promoted by social media. Moreover, popularity forecasting is a multistage rather than a single-stage task because each video may be propagated through a cascaded social network for a relatively long time and thus, the forecast can be made at any time while the video is being propagated. A fast prediction has important economic and technological benefits; however, too early a prediction may lead to a low accuracy that is less useful or even damaging (e.g., investment in videos that will not actually become popular). The timeliness of the prediction has yet to be considered in existing works that solely focus on maximizing the accuracy. Hence, developing a systematic methodology for accurate and timely popularity forecasting is essential.

26.1.4 OVERVIEW OF THE CHAPTER

A multiagent learning method that promotes cooperation and enables formation of cooperation networks between the agents is described in Section 26.2. How decentralized agents learn in a system that provides global or group feedback about the set of actions taken by the agents is discussed in Section 26.3. Learning and strategic network formation in networks with incomplete information is reviewed in Section 26.4. Concluding remarks are given in Section 26.5.

26.1.5 NOTATION

Unless otherwise stated, sets are denoted using calligraphic letters and vectors are denoted using boldface letters. $\mathcal{N}$ denotes the set of agents, $\mathcal{N}_i$ denotes the set of agents excluding agent i, $\mathcal{F}_i$ denotes the set of actions of agent i, $\mathcal{F}$ denotes the set of all actions, and $\mathcal{X}$ denotes the d-dimensional context set.

26.2 COOPERATIVE CONTEXTUAL BANDITS

In this section, we consider a multiagent decision-making and learning problem in which a set of cooperative and decentralized agents $\mathcal{N} = \{1, 2, \ldots, N\}$ help each other by selecting actions on behalf of each other to maximize their long-term rewards. In this model, each agent sequentially observes over time side information (context) about the environment that determines the expected rewards of the actions. Then, the agent decides to take an action or asks another agent to take an action on its behalf. If the agent chooses the latter option, then it is called the *requesting agent* and the agent that it asks to is called the *serving agent*. After this interaction both the requesting agent and the serving agent observe the random reward of the selected action, but the requesting agent may not observe the action selected by the serving agent. Such a situation happens in many real-world applications including real-time stream mining, where the agents may not be willing to reveal the type of classifiers they use to make predictions; network security, where agents may not be willing to reveal the number and type of the security protocols they use; and multimedia content aggregation, where an MCA does not have direct access to the content that is outside its content network. Both the requesting and serving agents benefit from this cooperation in the following way: the requesting agent makes use of the actions of the serving agent to obtain a high reward while the serving agent gets to know more about the rewards of its actions by taking an action for the requesting agent. For instance, in medical expertise discovery, the requesting institution benefits from the expertise of the expert of the serving institution by getting a more accurate diagnosis for its patient while the serving institution benefits from the experience gained through serving the other institution's patient, which may help it to match its patients with better experts in the future.

26.2.1 MODELING MULTIAGENT LEARNING USING COOPERATIVE CONTEXTUAL BANDITS

The set of serving agents of agent i is $\mathcal{N}_i := \mathcal{N} - \{i\}$, the set of actions of agent i is $\mathcal{F}_i$, and the set of all actions is $\mathcal{F} := \cup_{i\in\mathcal{N}}\mathcal{F}_i$. The set of choices of agent i is given as $\mathcal{K}_i := \mathcal{N}_i \cup \mathcal{F}_i$, which includes both its actions and its serving agents. Cardinalities of the sets $\mathcal{N}_i$, $\mathcal{F}_i$, and $\mathcal{K}_i$ are denoted by N_i, F_i, and K_i, respectively. An agent's action set is its private information. All agents know an upper bound on the number of actions that each agent has, which is denoted by $F_{\max}$. The system operates in discrete time where the following sequence of events take place at time t for all agents $i \in \mathcal{N}$:

- A d-dimensional context $x_i(t)$ arrives to agent i.
- Agent i selects a choice $a_i(t) \in \mathcal{K}_i$.
- If $a_i(t) \in \mathcal{F}_i$ agent i takes action $a_i(t)$.
- Else if $a_i(t) \in \mathcal{N}_i$, it sends $x_i(t)$ to agent $a_i(t)$, agent $a_i(t)$ takes an action in $\mathcal{F}_{a_i(t)}$ for agent i, and observes its reward.
- Agent i observes and collects the reward of the taken action.

The context set, denoted by $\mathcal{X}$, is assumed to be a bounded d-dimensional subset of the d-dimensional Euclidean space, which without loss of generality can be taken as $[0,1]^d$. The expected reward of action f for context x is $\pi_f(x)$. Neither $\pi_f(x)$ nor its distribution is known by any of the agents. We impose a natural assumption that relates the expected rewards of an action under different contexts, which is called the *similarity assumption*.

Similarity Assumption: $\exists L > 0, \alpha > 0$ such that $\forall f \in \mathcal{F}, \forall x, x' \in \mathcal{X}$

$$|\pi_f(x) - \pi_f(x')| \le L||x - x'||^{\alpha}, \tag{26.1}$$

where L and α determine the strength of similarity between the expected rewards of an action under two different contexts. It is assumed that this similarity information is known by the agents. For instance, in real-time stream mining the classification accuracy of a classifier may be similar for data streams with similar metadata, or in the medical expert assignment problem, the accuracy of the diagnosis of an expert may be similar for patients with similar features. While variants of the similarity measure defined in Eq. (26.1) are commonly used in the online learning literature [20–22], there are many other similarity measures that can be used, including similarity measures defined for categorical or mixed contexts [23,24].

In this setup, the goal of each agent is to maximize its total expected reward. Thus, each agent needs to learn the best choices for its contexts. However, the agents will not be able to achieve the maximum possible total expected reward due to the fact that they do not know the reward distributions of the actions beforehand. This loss due to learning is captured using the notion of *regret*, which measures how bad an agent performs with respect to an oracle that knows the set of all actions and their expected rewards. The set of the best actions of agent i for context x is $f_i^*(x) := \arg\max_{f\in\mathcal{F}_i}\pi_f(x)$. Hence, the maximum expected reward of agent i for context x is $\pi_i(x) = \pi_{f_i^*(x)}(x)$.[1,2] Based on this, the set of the

[1]With an abuse of notation, for a set g for which $\pi_h(x) = \pi_l(x)$ for all h and l in g, we use $\pi_g(x)$ to denote the expected reward of an element of g.

[2]We also use $k_i^*(x), f_i^*(x)$, and $f^*(x)$ to refer to an element of the corresponding set when the corresponding set contains more than one element.

best choices of agent i for context x is given as $k_i^*(x) := \arg\max_{k \in \mathcal{K}_i} \pi_k(x)$. On the other hand, the set of the best actions for context x is $f^*(x) := \arg\max_{f \in \mathcal{F}} \pi_f(x)$. Note that the expected reward of the best choice of agent i and the best action can be different when $k_i^*(x) \in \mathcal{N}_i$. This is due to the fact that the expected reward of $k_i^*(x)$ depends on the action chosen by agent $k_i^*(x)$. However, the maximum expected reward of agent $k_i^*(x)$ is equal to the expected reward of $f^*(x)$ in this case. Based on the definitions given above, the regret of agent i by time T is given as

$$R_i(T) := \sum_{t=1}^{T} \pi_{k_i^*(x_i(t))}(x_i(t)) - \mathrm{E}\left[\sum_{t=1}^{T} r^i_{a_i(t)}(x_i(t), t)\right], \tag{26.2}$$

where $r^i_{a_i(t)}(x_i(t), t)$ denotes the random reward of choice $a_i(t)$ of agent i at time t. Note that $r^i_{a_i(t)}(x_i(t), t)$ is a sample obtained from the reward distribution of action $a_i(t)$ when $a_i(t) \in \mathcal{F}_i$, and a sample obtained from the reward distribution of the action selected by agent $a_i(t)$ when $a_i(t) \in \mathcal{N}_i$.

The regret has two important properties: (i) it is nondecreasing in T, (ii) $R_i(T)/T$ gives the difference between the average rewards collected by the oracle and agent i. Therefore, when $R_i(T) = O(T^\gamma)$ for $\gamma < 1$ (i.e., the regret is sublinear), agent i is average reward optimal.

26.2.2 A DISTRIBUTED LEARNING ALGORITHM FOR COOPERATIVE CONTEXTUAL BANDITS

The online learning method used in cooperative contextual bandits consists of three essential components: a component that enables learning the expected rewards of the choices together for similar contexts, a component that enables agent i to learn the expected rewards of its actions and the maximum expected rewards of the set of serving agents accurately, and a component that enables agent i to maximize its total reward using what it had learned about its choices.

Because there are many contexts, estimating the expected reward of a choice for each context is in general infeasible. As a simple example, consider the case when agent i observes $x_i(t)$ for the first time, but has observed contexts that lie within the ϵ neighborhood of $x_i(t)$ many times. Without using the observations from the ϵ neighborhood of $x_i(t)$, the agent cannot do better than random guessing. On the other hand, the agent can get sufficiently accurate estimates of the expected rewards of its choices for context $x_i(t)$ by using the sample mean rewards of the choices calculated using the contexts within the ϵ neighborhood of $x_i(t)$. Thus, it is essential to form a partition of the context set $\mathcal{X}$ that is formed by sets that include similar contexts, and estimate the expected rewards of the choices for each context using the past observations of actions and rewards from the set to which the context belongs.

Two types of error must be taken into account when forming such a partition. The first type, called *dissimilarity error*, is due to the variation of the expected rewards of the contexts that lie in each set of the partition. For each set, this variation can be bounded using the diameter, i.e., the maximum distance between two contexts, of the set and the similarity assumption. The second type, called *sample size error*, is due to the limited number of random reward observations that are used to estimate the expected choice rewards. These two errors conflict with each other. For instance, decreasing the diameter of a set to reduce its dissimilarity error generally increases its sample size error because it corresponds to decreasing the number of past observations within the set. On the other hand, increasing the diameter of a set to reduce its sample size error generally increases the dissimilarity error. Thus, a balance between these two types of errors must be achieved in order to minimize the regret.

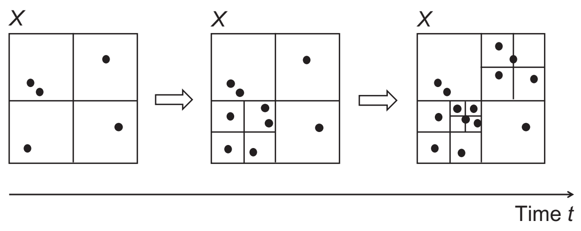

FIG. 26.1

An illustration of the contextual zooming method.

A natural way to achieve this balance is to use a technique called *contextual zooming* [21]. The idea is to adaptively form the partition of the context set based on the contexts that have arrived so far in a way that balances the dissimilarity error and the sample size error in each set of the partition. This implies that the partition will include many small sets concentrated around the region of the context set with high number context arrivals. An illustration of the contextual zooming method is shown in Fig. 26.1. Contextual zooming can be achieved by the following mechanism: Initially, agent i starts with partition $\mathcal{P}_i(1) = \{\mathcal{X}\}$, which only contains a single set. Let $\mathcal{P}_i(t)$ denote agent i's partition at time t, which consists of d-dimensional hypercubes with edge lengths coming from the set $\{2^0, 2^{-1}, 2^{-2}, \ldots\}$. A hypercube with edge length 2^{-l} is called a level-l hypercube, and for $p \in \mathcal{P}_i(t)$, $l(p)$ denotes its level. Also let $T^i_p(t)$ denote the number of contexts arrivals to $p \in \mathcal{P}_i(t)$ by time t, and $p_i(t)$ denote the hypercube in $\mathcal{P}_i(t)$ that contains context $x_i(t)$. $\mathcal{P}_i(t+1)$ is formed as follows:

- If $T^i_{p_i(t)}(t) \geq 2^{\rho l(p_i(t))}$, where $\rho > 0$ is a scale parameter, divide $p_i(t)$ into 2^d level $l(p_i(t)) + 1$ hypercubes. $\mathcal{P}_i(t+1)$ will contain all the sets in $\mathcal{P}_i(t)$ except $p_i(t)$ and will also contain the newly created level $l(p_i(t)) + 1$ hypercubes.
- Else $\mathcal{P}_i(t+1)$ will be the same as $\mathcal{P}_i(t)$.

The above procedure allows the *lifetime* of a hypercube to be an exponential function of its level. This bounds the number of hypercubes crated by time T as a sublinear function of T, and allows the agent to incur a small amount of regret in each hypercube such that when summed up over all hypercubes it still remains sublinear in T.

Next, we discuss how an agent learns the expected rewards of its choices accurately and uses them to maximize its total reward. We would like to emphasize that the agents cooperate with each other and obey the rules of the learning algorithm to choose actions on behalf of the other agents. Hence, cooperative contextual bandits do not consider strategic interactions between selfish agents. The agent maximizes its total reward by a three-phased mechanism that consists of *training*, *exploration*, and *exploitation* phases. For each context that arrives to agent i at time t, it can only be in one of these phases. Each phase is described below:

- **Training phase:** In this phase agent i trains its serving agents in $\mathcal{N}_i$. Training is done by choosing an *undertrained* agent in $\mathcal{N}_i$ as the serving agent. This agent j will receive $x_i(t)$, select an action $b_{j,i}(t) \in \mathcal{F}_j$, and observe its reward. As a result, it will update the estimated expected reward for its action $b_{j,i}(t)$. However, agent i is not going to update the estimated maximum reward for agent j

because the action selected by agent j is probably not its best action due to the fact that agent j is undertrained. The set of undertrained agents for agent i is given as

$$\mathcal{N}_i^{\mathrm{ut}}(t) := \left\{ j \in \mathcal{N}_i : N_{j,p_i(t)}^{\mathrm{tr},i}(t) \le D_{\mathrm{ut}}(p_i(t), t) \right\},$$

where $N_{j,p_i(t)}^{\mathrm{tr},i}(t)$ denotes the number of times agent i trained agent j for contexts in $p_i(t)$, and $D_{\mathrm{ut}}(p_i(t), t)$ is a control function that denotes the sufficient amount of trainings by time t that will enable agent j to form accurate estimates of the expected rewards of its own actions for $p_i(t)$.

- **Exploration phase:** In this phase agent i selects an *underexplored* choice in $\mathcal{K}_i$, and uses the reward it obtains from its selection to update the estimated expected reward of its choice. Contrary to the training phase, in the exploration phase the action chosen by the serving agent is its best action with a high probability. The set of underexplored choices of agent i is given as

$$\mathcal{N}_i^{\mathrm{ue}}(t) := \left\{ j \in \mathcal{K}_i : N_{j,p_i(t)}^{\mathrm{ue},i}(t) \le D_{\mathrm{ue}}(p_i(t), t) \right\},$$

where $N_{j,p_i(t)}^{\mathrm{ue},i}(t)$ denotes the number of times choice j is selected by agent i except selections in the training phases of agent i for contexts in $p_i(t)$, by time t, and $D_{\mathrm{ue}}(p_i(t), t)$ is a control function that denotes the sufficient amount of explorations by time t that will enable agent i to form accurate estimates of the expected rewards of its choices for $p_i(t)$.
- **Exploitation phase:** In this phase agent i selects the choice with the highest estimated expected reward in order to maximize its long-term reward.

When agent i is selecting a choice for its own context at time t, it can be in three of the above phases. If there are any underexplored actions of agent i, it will be in the exploration phase. Else if there are any undertrained agents of agent i, it will be in the training phase. Else if there are any underexplored agents of agent i, it will be in the exploration phase. Else, it will be in the exploitation phase. On the other hand, as a serving agent, agent i can either be in the exploration or the exploitation phase because it selects an action from $\mathcal{F}_i$. Due to the heterogeneity of the context arrivals, usually an agent cannot learn the expected rewards of its actions well for all possible contexts. However, cooperation enables the agents to learn the expected rewards of their actions for contexts that are observed frequently by the other agents. An agent can benefit from this learning in the future if the future contexts are not among contexts that are frequently observed by the agent thus far. On the other hand, the requesting agent gets an immediate benefit in terms of the reward through using the actions of the serving agents.

Next, we provide a result on the performance of the distributed learning algorithm discussed in this section [20].

Theorem 26.1. *Consider the distributed learning algorithm for the cooperative contextual bandit problem. Assuming that the rewards of the actions lie in* $[0, 1]$, *there exists a set of parameters* ρ, $D_{ut}(p, t)$ *and* $D_{ue}(p, t)$ *for any* p *such that*

$$R_i(T) = \tilde{O}(T^{\frac{2\alpha+d}{3\alpha+d}})$$

for any sequence of context arrivals $x_i(1), x_i(2), \ldots, x_i(T)$ *and for all* $i \in \mathcal{N}$. ■

This theorem shows that the regret of agent i is sublinear in time. The actions selected in training and exploration phases can be suboptimal. Therefore, the number of training and exploration phases must be sublinear in order to achieve sublinear regret. This implies that the agent exploits for the majority of the time when T is large. The choices selected by the agent when it exploits form its cooperation network.

The ideal cooperation network of agent i can be computed if $\pi_k(x)$ is perfectly known $\forall x \in \mathcal{X}$ and $\forall k \in \mathcal{K}_i$. The region of optimality of a choice $k \in \mathcal{K}_i$ can be defined as $\mathcal{X}^*_{i,k} := \{x \in \mathcal{X} : k \in k^*_i(x)\}$. $\mathcal{X}^*_{i,k} = \emptyset$, implies that choice k is never optimal for agent i. The ideal cooperation network of agent i consists of other agents that are optimal choices for agent i for at least one context in $\mathcal{X}$. Hence, the ideal cooperation network of agent i is given as $\mathcal{N}^*_i := \{k \in \mathcal{N}_i : \mathcal{X}^*_{i,k} \neq \emptyset\}$. Although an agent might be in the ideal cooperation network of agent i, it may not be optimal for a particular realization $\boldsymbol{x}_i := \{x_i(1), \dots, x_i(T)\}$ of agent i's contexts. We can define the realization-specific ideal cooperation network of agent i as the set of other agents who are optimal choices for agent i for at least one context in $x_i(1), \dots, x_i(T)$, i.e., $\mathcal{R}^*_i(\boldsymbol{x}_i) := \{k \in \mathcal{N}_i : k \in k^*_i(x) \text{ for some } x \in \{x_i(1), \dots, x_i(T)\}\}$. Obviously, we have $\mathcal{R}^*_i(\boldsymbol{x}_i) \subseteq \mathcal{N}^*_i$.

On the other hand, the real cooperation network of agent i at time t consists of other agents that it is willing to cooperate with based on their estimated expected rewards. Let $\hat{\pi}^i_{k,p}(t)$ denote the estimated expected reward of choice k for contexts in $p \in \mathcal{P}_i(t)$ formed by agent i at time t. This can be taken as the sample mean of the rewards observed for choice k from contexts in p excluding the observations from the training phases of agent i. Based on this, the set of estimated optimal choices of agent i at time t for $p \in \mathcal{P}_i(t)$ is given as $\hat{k}^*_{i,p}(t) := \arg\max_{k \in \mathcal{K}_i} \hat{\pi}^i_{k,p}(t)$. Note that $a_i(t) \in \hat{k}^*_{i,p}(t)$ when agent i exploits at time t. Hence, the real cooperation network of agent i at time t is given as $\hat{\mathcal{N}}^*_i(t) := \{k \in \mathcal{N}_i : k \in \hat{k}^*_{i,p}(t) \text{ for some } p \in \mathcal{P}_i(t)\}$. The real cooperation network is expected to converge to the ideal cooperation network as the estimated expected rewards of the choices for all the contexts converge to the true expected rewards. Moreover, the realized ideal cooperation network is expected to converge to the ideal cooperation network when for all $k \in \mathcal{N}^*_i$, there exists at least one $x_i(t)$ in $\{x_i(1), \dots, x_i(T)\}$ such that $x_i(t) \in \mathcal{X}^*_{i,k}$.

26.2.3 LEARNING TO COOPERATE WHEN COOPERATION IS COSTLY

Thus far we assumed that agents can cooperate with each other freely without incurring any costs. An interesting extension is to force the requesting agent i to incur a penalty, denoted by $d^i_k > 0$ every time it chooses a serving agent k. For instance, in a network security application the agents will correspond to *autonomous systems* (ASs) that collaborate with each other to detect cyberattacks. This requires exchanging network traffic information between the ASs, which can incur both communication cost and cost related to using the resources of the serving AS to detect the cyberattack. Another interesting example is expert selection for medical diagnosis described in Section 26.1.2. In this example, actions correspond to medical experts that will diagnose a patient and agents correspond to clinics. It is possible that a clinic refers the patient to another clinic, and the serving clinic charges the requesting clinic for the service. A similar situation may also arise in the multimedia content aggregation application described in Section 26.1.1.

There are numerous ways to tackle cooperation costs. A straightforward way is to define the expected payoff of each choice $k \in \mathcal{N}_i$ for context x as $\pi_k(x) - \gamma d^i_k$, where $\gamma > 0$ is a weight that defines the tradeoff between the reward and the cost. Then, agent i can run the distributed learning algorithm described in Section 26.2.2 by using estimated expected payoffs for the choices $k \in \mathcal{N}_i$, i.e., $\hat{\mu}^i_{k,p}(t) := \hat{\pi}^i_{k,p}(t) - \gamma d^i_k$. Because this modification also changes the oracle, the definition of the regret in Eq. (26.2) and the regret bound in Theorem 26.1 remains unchanged. However, the ideal and real cooperation networks of the agents will be affected by the introduction of cooperation costs. When the cooperation costs are high, agents will learn to cooperate with the other agents only if the expected payoff from the cooperation exceeds the expected reward that the agent can obtain from its own actions. This implies that the region of optimality for a choice $k \in \mathcal{N}_i$ is expected to shrink from $\mathcal{X}^*_{i,k}$ to $\emptyset$ as the cooperation cost d^i_k increases from 0 to ∞.

The cooperation networks that are formed by the agents can vary based on how the tradeoff between the reward and cooperation cost is captured. Another way to capture this tradeoff is to consider cooperative contextual bandits as a multiobjective online learning problem. Then, the reward can be taken as one objective and the cost can be taken as the other objective.

In the multiobjective online learning problem, the distributed learning algorithm can be modified to learn the Pareto front. In this case, it is natural to define the regret as the sum of the distances of the actions chosen by the agents to the Pareto front [25]. In addition, since the Pareto front depends on the context, the contextual zooming method should be adapted to learn the choices that are in the Pareto front [26].

Another alternative is to consider specific subsets of the Pareto front. One interesting case is when one objective dominates the other objective. For instance, the agents may want to select choices that maximize their reward, and if there are multiple such choices, they may also want to select choices that minimize the cost among all choices that maximize the reward. In this case, the reward dominates the cost, and hence, the cooperative contextual bandit problem can be modeled as a multiobjective contextual bandit problem with a dominant objective [27].

26.3 DISTRIBUTED LEARNING WITH GLOBAL OR GROUP FEEDBACK

In this section, we consider a multiagent decision-making and learning problem in which a set of distributed agents $\mathcal{N} = \{1, 2, \ldots, N\}$ select actions from their own action sets $\mathcal{A}_n$, with the cardinality of the action set denoted by $K_n = |A_n|, \forall n \in \mathcal{N}$ in order to maximize the overall system reward $r(\boldsymbol{a})$. The system reward depends on the joint action $\boldsymbol{a} = (a_1, \ldots, a_N)$ of all agents but agents do not know *a priori* how their actions influence the overall system reward, or how their influence may change dynamically over time. Each time, agents can only observe/measure the *overall* system performance and hence, they only obtain global feedback that depends on the joint actions of all agents, yet the observations of the global feedback may be subject to individual errors. That is, each agent n privately observes $r^n_t = r_t + \epsilon^n_t$, equal to the global reward r_t plus a random noise term ϵ^n_t. The fact that individualized feedback is missing, communication is not possible, and the global feedback is noisy makes the development of efficient learning algorithms that maximize the joint reward very challenging. Fig. 26.2 portrays distributed cooperative learning with individualized feedback and global feedback with individual noises.

The considered problem has many application scenarios. For instance, in a stream-mining system that uses multiple classifiers for event detection in video streams, individual classifiers select operating points to classify specific objects or actions of interest, the results of which are synthesized to derive an event classification result (see Fig. 26.3). If the global feedback is only about whether the event classification is correct, individualized feedback about individual contribution is not available. For another instance, in a cooperative communication system, a set of wireless nodes forwards signals of

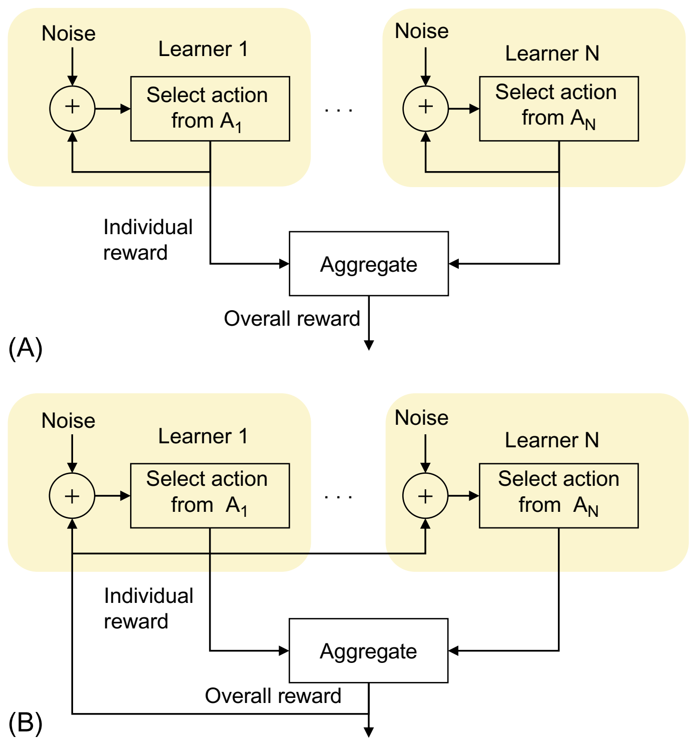

FIG. 26.2

(A) Individualized feedback; (B) Global feedback.

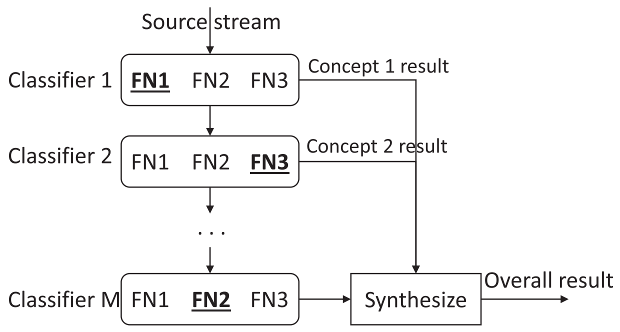

FIG. 26.3

Classifier chain for real-time stream mining.

the same message copy to a destination node through noisy wireless channels. Each forwarding node selects its transmission scheme (e.g., power level) and the destination combines the forwarded signals to decode the original message using, e.g., a maximal ratio combination scheme. Because the message is only decoded using the combined signal but not individual signals, only a global reward depending on the joint effort of the forwarding nodes is available but not the nodes' individual contributions.

The rewards in the considered problem are highly correlated among agents and hence, it is important to design algorithms that take this into account. An interesting literature along this line is combinatorial bandit [28], in which the goal is to exploit the correlations between the rewards. In this problem, the agent chooses an action vector and receives a reward that depends on some linear or nonlinear combination of the individual rewards of the actions. In a combinatorial bandit problem the set of arms grows exponentially with the dimension of the action vector; thus standard bandit policies such as the one in [29] will have a large regret. The idea in these problems is to exploit the correlations between the rewards of different arms to improve the learning rate and thereby reduce the regret [30,31]. Most of the works on combinatorial bandits assume that the expected reward of an arm is a linear function of the chosen actions for that arm. For example, [32] assumes that after an action vector is selected, the individual rewards for each nonzero element of the action vector are revealed. Another work [33] considers combinatorial bandit problems with more general reward functions, defines the approximation regret, and shows that it grows logarithmically in time. The approximation regret compares the performance of the learning algorithm with an oracle that acts approximately optimally while we compare our algorithm with the optimal policy. This work also assumes that individual observations are available. However, in this paper we assume that only global feedback is available and individuals cannot observe each other's actions. Agents have to learn their optimal actions based only on the feedback about the overall reward. Other bandit problems that use linear reward models are studied in [34–36]. These consider the case where only the overall reward of the action profile is revealed but not the individual rewards of each action. However, our analysis is not restricted to linear reward models, but instead are much more general.

Another line of work considers online optimization problems where the goal is to minimize the loss due to learning the optimal vector of actions that maximizes the expected reward function. These works show sublinear (but greater than logarithmic) regret bounds for linear or submodular expected reward functions when the rewards are generated by an adversary to minimize the gain of the agent. The difference of our work is that we consider a more general reward function and prove logarithmic regret bounds. Recently, distributed bandit algorithms were developed in [37] in network settings. In that work, agents have the same set of arms with the same unknown distributions and are allowed to communicate with neighbors to share their observed rewards. In contrast, agents in our problem have distinct sets of arms, the reward depends on the joint action of agents, and agents do not communicate at run-time.

26.3.1 ACHIEVING COOPERATION THROUGH GLOBAL FEEDBACK

We propose multiagent learning algorithms that enable the various agents to learn how to make decisions to maximize the overall system reward without exchanging information with other agents. In order to quantify the loss due to learning and operating in an unknown environment, we define the regret of an online learning algorithm for the set of agents as the difference between the expected reward of the best joint action of all agents and the expected reward of the algorithm used by the agents. We prove that, if the global feedback is received without errors by the agents, then all deterministic algorithms can be implemented in a distributed manner without message exchanges. This implies that the distributed nature of the system does not introduce any performance loss compared with a centralized system because there exist deterministic algorithms that are optimal. Subsequently, we show that if agents receive the global feedback *with* different (individual) errors, existing deterministic algorithms may break down and hence, there is a need for novel distributed cooperative algorithms that are robust to such errors. For this, we develop a class of algorithms that achieves a logarithmic upper bound on the regret, implying that the average reward converges to the optimal average reward. The upper bound on regret also gives a lower bound on the convergence rate to the optimal average reward.

We start with a basic *distributed cooperative learning* (DisCo) algorithm without making any additional assumptions on the reward structure. The DisCo algorithm is divided into phases: exploration and exploitation. Each agent using DisCo will alternate between these two phases in a way that at any time t, either all agents are exploring or all are exploiting. In the exploration phase, each agent selects an arm only to learn about the effects on the expected reward, without considering reward maximization, and updates the reward estimates of the arm it selected. In the exploitation phase, each agent exploits the best (estimated) arm to maximize the overall reward. There is a deterministic control function $\zeta(t)$ of the form $\zeta(t) = A \ln t$ commonly known by all agents. This function will be designed and determined before run-time, and thus is an input of the algorithm. Each exploration phase has a fixed length of $L_1 = \prod_{n=1}^{N} K_n$ slots, equal to the total number of joint arms. Each agent maintains two counters. The first counter $\gamma(t)$ records the number of exploration phases that they have experienced by time slot t. The second counter $E(t) \in \{0, 1, \ldots, L_1\}$ represents whether the current slot is an exploration slot and, if yes, which relative position it is at. Specifically, $E(t) = 0$ means that the current slot is an exploitation slot; $E(t) > 0$ means that the current slot is the $E(t)$th slot in the current exploration phase. Both counters are initialized to zero: $\gamma(0) = 0, E(0) = 0$. Each agent n maintains L_1 sample mean reward estimates $\bar{r}^n(l)\ \forall l \in \{1, \ldots, L_1\}$, one for each relative slot position in an exploration phase. Let b_l^n denote the arm selected by agent n in the lth position in an exploration phase. These reward estimates are initialized to be $\bar{r}^n(l) = 0$ and will be updated over time using the realized rewards. Whether a new slot t is an exploration slot or an exploitation slot will be determined by the values of $\zeta(t)$, $\gamma(t)$ and $E(t)$. At the beginning of each slot t, the agents first check the counter $E(t)$ to see whether they are still in the exploration phase: if $E(t) > 0$, then the slot is an exploration slot; if $E(t) = 0$, whether the slot is an exploration slot or an exploitation slot will then be determined by $\gamma(t)$ and $\zeta(t)$. If $\gamma(t) \le \zeta(t)$, then the agents start a new exploration phase, and at this point $E(t)$ is set to be $E(t) = 1$. If $\gamma(t) > \zeta(t)$, then the slot is an exploitation slot. At the end of each exploration slot, counter $E(t+1)$ for the next slot is updated to be $E(t+1) \leftarrow mod(E(t)+1, L_1+1)$. When $E(t+1) = 0$, the current exploration phase ends, and hence the counter $\gamma(t+1)$ for the next slot is updated to be $\gamma(t+1) \leftarrow \gamma(t)+1$. Fig. 26.4 provides the flowchart of the phase transition for the algorithm. The algorithm prescribes different actions for agents in different slots and in different phases.

(i) *Exploration phase*: As is clear from above, an exploration phase consists of L_1 slots. In each phase, the agents select their own arms in such a way that every joint arm is selected exactly once. This is possible without communication if agents agree on a selection order for the joint arms before run-time. At the end of each exploration slot (the lth slot), $\bar{r}^n(l)$ is updated to

$$\bar{r}^n(l) \leftarrow \frac{(\gamma(t) - 1)\bar{r}^n(l) + r_t^n}{\gamma(t)}. \tag{26.3}$$

Note that the observed reward realization r_t^n at time t may be different for different agents due to errors.

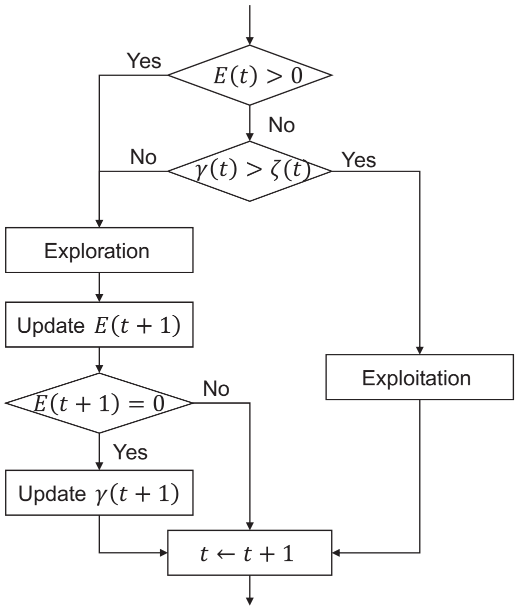

FIG. 26.4

Flowchart of the phase transition.

(ii) *Exploitation phase*: Each exploitation phase has a variable length that depends on the control function $\zeta(t)$ and counter $\gamma(t)$. At each exploitation slot t, each agent n selects $a_n = \{b^n_{l^*} : l^* = \arg\max_l \bar{r}^n(l)\}$. That is, each agent n selects the arm with the best reward estimate among $\bar{r}^n(l)$, $\forall l \in \{1, \ldots, L_1\}$. Note that in the exploitation slots, an agent n does not need to know other agents' selected arms. Because agents have individual observation noises, it is also possible that l^* is different for different agents.

Regret of the DisCo algorithm is bounded in the following theorem [38].

Theorem 26.2. *If $\zeta(t) = A \ln t$ with $A > 2\left(\frac{D}{\Delta^{\min}}\right)^2$, then the expected regret of the DisCo algorithm after any number T periods is bounded by*

$$R(T) \leq AL_1 \Delta^{\max} \ln T + B_1, \tag{26.4}$$

where $B_1 = L_1 \Delta^{\max} + \sum_{t=1}^{\infty} 2NL_1 \Delta^{\max} t^{-\frac{A}{2}\left(\frac{\Delta^{\min}}{D}\right)^2}$ is a constant. Here $\Delta^{\max}$ represents the difference between the expected rewards of the best joint arm and the second-best joint arm, and $\Delta^{\min}$ represents the maximum expected loss due to selecting any suboptimal joint arm. ■

26.3.2 ACCELERATING LEARNING THROUGH REWARD INFORMATIVENESS

Although the time order of the regret of the DisCo algorithm is logarithmic, due to its linear dependence on the cardinality of the joint action space, which increases exponentially with the number of agents, the regret is large and the convergence rate is very slow with many agents. In many application scenarios, even if we do not know exactly how the actions of agents determine the expected overall rewards, some structural properties of the overall reward function may be known. For example, in the classification problem that uses multiple classifiers [39], the overall classification accuracy is increasing in each individual classifier's accuracy, even though each individual's optimal action is unknown a priori. Thus, some overall reward functions may provide higher levels of informativeness about the optimality of individual actions. Therefore, we also develop learning algorithms that achieve faster learning speed by exploiting such information.

Definition 26.1. An expected overall reward function $\mu(\boldsymbol{a})$ is said to be informative with respect to agent n if there exists a unique arm $a_n^* \in \mathcal{A}_n$ such that $\forall \boldsymbol{a}_{-n}$, $a_n^* = \arg\max_{a_n} \mu(a_n, \boldsymbol{a}_{-n})$, where $\mu(\boldsymbol{a})$ represents the expected reward of a joint arm $\boldsymbol{a}$. An expected overall reward function $\mu(\boldsymbol{a})$ is said to be fully informative if it is informative with respect to all agents. ■

If the overall reward function is fully informative, then the agents only need to record the relative overall reward estimates instead of the exact overall reward estimates. Therefore, the key design problem of the learning algorithm is, for each agent n, to ensure that the fraction of time selecting $\boldsymbol{a}_{-n}$ (called weight $\theta_{\boldsymbol{a}_{-n}}$) is the same for the relative reward estimates for any given arm a_n so that it is sufficient for agent n to learn the optimal arm using only these relative reward estimates. We emphasize the importance of the weights $\theta_{\boldsymbol{a}_{-n}}, \forall \boldsymbol{a}_{-n}$ being the same for all $a_n \in \mathcal{A}_n$ of each agent n even though agent n does not need to know these weights exactly. If the weights are different for different a_n, then it is possible that $\bar{r}^n(a_n') > \bar{r}^n(a_n^*)$ merely because other agents are using their good arms when agent n is selecting a suboptimal arm a_n while other agents are using their bad arms when agent n is selecting the optimal arm a_n^*. Hence, simply relying on the relative reward estimates does not guarantee obtaining the correct information needed to find the optimal arm.

Next, we describe an improved learning algorithm. We call this new algorithm the DisCo-FI algorithm where "FI" stands for "Fully Informative." The key difference from the basic DisCo algorithm is that, in DisCo-FI, the agents will maintain relative reward estimates instead of the exact reward estimates. Agents know a common deterministic function $\zeta(t)$ and maintain two counters $\gamma(t)$ and $E(t)$. Now each exploration phase has a fixed length of $L_2 = \sum_{n=1}^{N} K_n$ slots and hence, $E(t) \in \{0, 1, \ldots, L_2\}$ with $E(t) = 0$ representing that the slot is an exploitation slot and $E(t) > 0$ representing that it is the $E(t)$th relative slot in the current exploration phase. As before, both counters are initialized to be $\gamma(0) = 0, E(0) = 0$. Each agent n maintains K_n sample mean (relative) reward estimates $\bar{r}^n(a_n), \forall a_n \in \mathcal{A}_n$, one for each one of its own arms. These (relative) reward estimates are initialized to be $\bar{r}^n(a_n) = 0$ and will be updated over time using the realized rewards. The transition between exploration phases and exploitation phases is almost identical to that in the DisCo algorithm. The only difference is that at the end of each exploration slot, the counter $E(t+1)$ for the next slot is updated to be $E(t+1) \leftarrow mod(E(t)+1, L_2+1)$. Hence, we ensure that each exploration phase has only L_2 slots. The algorithm prescribes different actions for agents in different slots and in different phases.

(i) Exploration phase: As is clear from the phase transition, an exploration phase consists of L_2 slots. These slots are further divided into N subphases and the length of the nth subphase is K_n. In the nth subphase, agents take actions as follows:

1. Agent n selects each of its arms $a_n \in \mathcal{A}_n$ in turn, each arm for one slot. At the end of each slot in this subphase, it updates its reward estimate using the realized reward in this slot as follows,

$$\bar{r}^n(a_n) \leftarrow \frac{\gamma(t)\bar{r}^n(a_n) + r_t^n}{\gamma(t) + 1}. \tag{26.5}$$

2. Agent $i \neq n$ selects the arm with the highest reward estimate for every slot in this subphase, i.e., $a_i(t) = \arg\max_{a_i \in \mathcal{A}_i} \bar{r}^i(a_i)$.

(ii) Exploitation phase: Each exploitation phase has a variable length that depends on the control function $\zeta(t)$ and counter $\gamma(t)$. In each exploitation slot t, each agent n selects $a_n(t) = \arg\max_{a \in \mathcal{A}_n} \bar{r}^n(a)$.

Theorem 26.3 ([38]). *Suppose $\mu(\boldsymbol{a})$ is fully informative. If $\zeta(t) = A \ln T$ with $A \geq 2 \left(\frac{D}{\Delta_{FI}^{\min}} \right)^2$, then the expected regret of the DisCo-FI algorithm after any number T slots is bounded by*

$$R(T) < AL_2 \Delta^{\max} \ln T + B_2, \tag{26.6}$$

where $B_2 = L_2 \Delta^{\max} + 2L_2 \Delta^{\max} \sum_{t=1}^{\infty} t^{-\frac{A}{2}\left(\frac{\Delta_{FI}^{\min}}{D}\right)^2}$ is a constant number. Here $\Delta_{FI}^{\min} = \min_n \Delta_n^{\min}$ where $\Delta_n^{\min}$ is the expected reward difference between agent n's best arm and its second-best arm. ■

DisCo-FI can be extended to the more general case where the full informativeness constraint is relaxed.

Definition 26.2. An expected overall reward function $\mu(\boldsymbol{a})$ is said to be informative with respect to a group of agents g_m if there exists a unique group-joint arm $\boldsymbol{a}_m^* \in \times_{n \in g_m} \mathcal{A}_n$ such that $\forall \boldsymbol{a}_{-m}, \boldsymbol{a}_m^* = \arg\max_{\boldsymbol{a}_m \in \times_{i \in g_m} \mathcal{A}_i} \mu(\boldsymbol{a}_m, \boldsymbol{a}_{-m})$. An expected overall reward function $\mu(a)$ is said to be partially informative with respect to a group partition $\mathcal{G} = \{g_1, \ldots, g_M\}$ if it is informative with respect to all groups in $\mathcal{G}$. ■

The new algorithm is called the DisCo-PI algorithm where "PI" stands for "Partially Informative." It is a combination of the basic DisCo and DisCo-FI. In particular, each agent group is treated as a single agent in the DisCo-FI algorithm and hence, each agent group learns the relative rewards of its group-joint arms. Within each group, agents follow the idea of the basic DisCo algorithm to learn their optimal arms.

Theorem 26.4. *[38] Suppose $\mu(\boldsymbol{a})$ is partially informative with respect to a group partition $\mathcal{G}$. If $\zeta(t) = A \ln T$ with $A \geq 2 \left(\frac{D}{\Delta_{PI}^{\min}} \right)^2$, then the expected regret of the DisCo-PI algorithm after any number T slots is bounded by*

$$R(T) < AL_3 \Delta^{\max} \ln T + B_3, \tag{26.7}$$

where $L_3 = \sum_{m=1}^{M} \prod_{n \in g_m} K_n$ and

$$B_3 = L_3 \Delta^{\max} + 2 \sum_{m=1}^{M} N_m \prod_{n \in g_m} K_n \Delta^{\max} \sum_{t=1}^{\infty} t^{-\frac{A}{2}\left(\frac{\Delta_{PI}^{\min}}{D}\right)^2} \tag{26.8}$$

is a constant number. Here $\Delta_{PI}^{\min} = \min_m \Delta_m^{\min}$ where $\Delta_m^{\min}$ is the expected reward difference between the best group-joint arm of g_m and the second-best group-joint arm of g_m. ■

We illustrate the performance of the proposed learning algorithms via simulation results for the real-time stream-mining problem using multiple classifiers in Fig. 26.5. The proposed algorithms are compared against several benchmarks: random, safe experimentation, UCB1 [29], and optimal. Safe experimentation (SE) is a method used in [40] when there is no uncertainty about the accuracy of the classifiers. In each period t, each classifier selects its baseline action with probability $1 - \epsilon_t$ or selects a new random action with probability ϵ_t. When the realized reward is higher than the baseline reward, the classifiers update their baseline actions to the new action. As can been seen, SE works almost as poorly as the random benchmark in terms of event detection accuracy. Due to the uncertainty in the

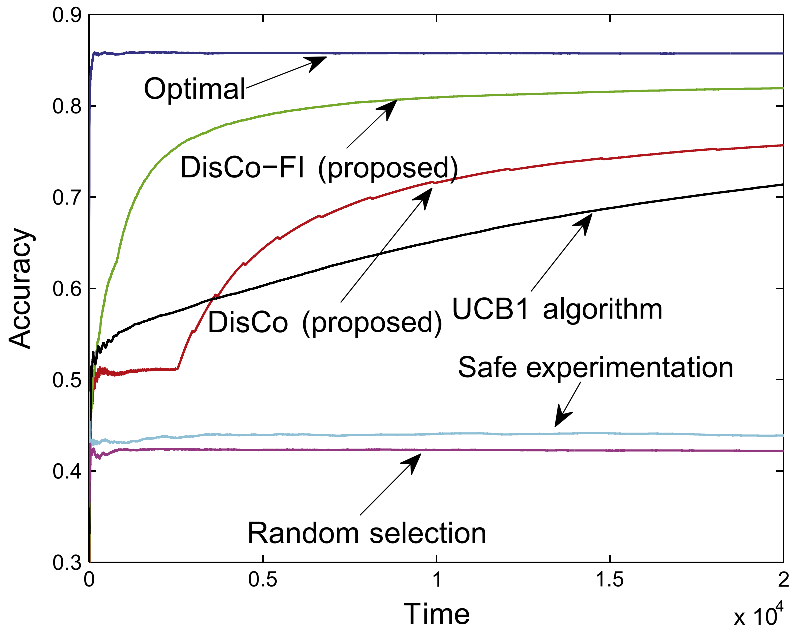

FIG. 26.5

Performance comparison [38].

detection results, updating the baseline action to a new action with a higher realized reward does not necessarily lead to selecting a better baseline action. Hence, SE is not able to learn the optimal operating points of the classifiers. UCB1 achieves a much higher accuracy than random and SE algorithms and is able to learn the optimal joint operating points over time. However, the learning speed is slow because the joint arm space is large. The proposed DisCo algorithm can also learn the optimal joint action. However, because the joint arm space is large, the classifiers have to stay in the exploration phases for a relatively long time in the initial periods to gain sufficiently high confidence in reward estimates while the exploitation phases are rare and short. Thus, the classification accuracy is low initially. After the initial exploration phases, the classifiers begin to exploit and hence the average accuracy increases rapidly. Because the reward structure satisfies the fully informative condition, DisCo-FI rapidly learns the optimal joint action and performs the best among all schemes.

26.4 LEARNING IN NETWORKS WITH INCOMPLETE INFORMATION

Learning plays a major role in the analysis of networks and network formation. Many significant real-world interactions take place over a variety of networks, such as social networks, financial networks, and trade networks. Learning and incomplete information can strongly affect the development and dynamics of such networks, and so it is important to properly model and study these forces in performing network analysis.

The existing literature has broken down learning in networks into two different types. The first type of learning is learning about an exogenous state of the world. Topics of study include how information among agents is aggregated and whether an accurate consensus in the network can eventually form. The second type of learning is learning about the qualities of other agents within the network. Papers in this literature analyze how agents choose their links under incomplete information, how learning about other agents affects the network development, and how efficient the networks that form from this process are. Both types of learning are of great importance, and both have been the subject of active research. This chapter will provide an overview of these two types of learning, describe the recent work that has been performed, and discuss the conclusions that such work has reached.

26.4.1 LEARNING AND OPINION DYNAMICS

In the real world, people frequently learn and exchange information with those they are closely connected to, such as friends, family, or coworkers. This information may regard a range of decisions both major and minor, such as which clothes to buy, which shows to watch, or where to live. Agents can learn information directly by communicating with their neighbors or indirectly by observing their neighbors' actions and choices. As time passes, agents may converge on a common belief through this process of mutual influence, and a consensus could emerge within the network.

Such situations have been analyzed within a group of recent papers in the network literature. In these papers, agents receive information directly from their neighbors or learn by observing the decisions their neighbors make. They then use this to update their own beliefs and make their own decisions. A key question that is raised is the issue of learning efficiency and the convergence of beliefs to the truth. If information is initially diffused across a network, is it possible for the information to be aggregated through a learning process such that the network arrives at the truth? Can this property be guaranteed if the network is sufficiently large or the agents sufficiently numerous? This question is important for understanding how learning occurs within very large networks, such as societies.

The existing literature has shown that the configuration of the network has a strong impact on the information that agents learn and the rate at which they can reach a consensus. Golub and Jackson [41] consider such a problem when agents utilize a form of learning known as DeGroot learning. This type of learning assumes that each period, agents update their belief based on the average beliefs of their neighbors, and can be interpreted, for instance, as agents communicating with and influencing each other during each period. The paper shows that the network converges to the correct belief if and only if no agent has too large an influence. If there is a prominent group of agents that receive too much attention, then learning will not converge to the truth.

Acemoglu et al. [42] consider a different type of opinion dynamics where the agents do not directly communicate their beliefs with others but merely observe other agents' actions. For instance, an agent may observe the car that another agent bought, and thus update its own belief more favorably for that car manufacturer. This is known in the general economic theory literature as social learning, which has its own extensive literature but was only recently applied to network theory. The paper shows that learning will be correct asymptotically as long as the observation network is large enough. This is satisfied in settings where agents are able to draw from enough observations.

While these other papers consider observations that occur in exogenously given networks, some other papers have allowed for endogenous network formation. Acemoglu et al. [43] consider a model in which agents form communication links with each other by paying a cost. Through these links agents can transmit information to each other. The authors show that learning converges to the truth if the formed network feature “information hubs” that can distribute information to the agents. Thus learning must be aggregated within a few agents. Furthermore, they show that if the network has too many social cliques that contain the spread of information, then learning may not converge to the truth.

Song [44] also allows for endogenous observations. In this model, agents make observations sequentially as in Acemoglu et al. [42], but they are able to choose which other agents to observe. Agents have a fixed capacity of observations to make, and thus they must be strategic in choosing which other agents to observe. Whether the eventual consensus in the network is accurate will depend on the specific links that agents choose to form. This model introduces important aspects of network formation, discussed further in the following section, into the social learning problem. The paper shows that under certain situations, there will indeed be equilibrium in which agents are able to correctly learn. In such equilibrium, agents observe a large enough group of other agents so that the agent signals can be aggregated correctly.

26.4.2 LEARNING AND STRATEGIC NETWORK FORMATION

The other major type of learning that has been analyzed within the network literature is learning about other agents through the process of strategic network formation. Whereas opinion dynamics models analyze situations in which agents do not directly derive benefits from their neighbors but instead learn information about an exogenous state of the world, strategic network formation models assume that agents directly derive benefits from their links with others. In forming a link, an agent weighs the benefits provided through that link versus the costs of maintaining the link. The beliefs an agent has about the benefits each neighbor can provide will strongly influence which neighbors the agent chooses to link with, and learning in turn will strongly affect the network structure and dynamics. In this section, we provide an overview of such models and summarize the main conclusions that have been reached.

Strategic network formation itself has developed over the past several decades into a large literature. However, many papers in this field have not considered the presence of incomplete information in their models. Seminal strategic network formation papers, such as Jackson and Wolinsky [45], Bala and Goyal [46], and Jackson and Watts [47], all assume that agents have complete information about others, and thus agents know exactly how much benefit they will receive from each link. Because information is perfect, agent beliefs are always accurate and never update over time.

Incorporating incomplete information into strategic network formation models represents an important step toward more accurately modeling real-world interactions. In many real-world networks, incomplete information often has a profound effect. For instance, in a social network agents who are meeting and making friends will begin with only limited information about each other. It takes mutual experiences to learn about others, and as such learning occurs the network structure will evolve.

A recent strand of network papers has developed to analyze the impact of incomplete information on the network formation process and the resulting network structures. These models operate by assuming agents have different types. Agents of different types may provide different payoffs to others, receive different payoffs from others, or both. Importantly, each agent’s type is unknown to others at the beginning of the model. Instead, agents have prior beliefs about other agent types. As links are formed, information is learned about a neighbor’s type through the benefits that are received, and this allows an agent to update its prior beliefs using the Bayes rule. The prior beliefs can have a strong impact on the eventual stable network, however, as they will affect the initial links that are formed, which influences subsequent learning. Thus, the initial beliefs can have lasting consequences for the network.

These models demonstrate that incomplete information can have a strong impact on the network formation process. First, incomplete information will affect the initial links that are formed among the agents. Even if an agent has a high true quality, if others believe that the agent's quality is low, then they will never form a link with that agent. Thus, it is possible for some high-quality agents to be excluded immediately from the network. This presents a source of inefficiency from the incomplete information.

Incomplete information may provide social benefits as well. If links have externalities, then it may be socially beneficial for agents to form links to bad agents. For instance, a common payoff model within the network literature is the connections model, in which agents receive benefits from direct neighbors as well as indirect benefits from neighbors of direct neighbors. Even if a direct neighbor's quality is low, if that direct neighbor has many indirect neighbors it may be beneficial to form a link to it. There can be multiple stable networks with varying levels of indirect benefits provided, and the socially optimal network may not be reachable along certain formation paths. Incomplete information can increase the amount of formation paths, and give a chance for the optimal network to develop. The paper by Song and van der Schaar [48] highlights theses effects via a discrete-time network formation model. They show that a large increase in potential networks arises under incomplete information, and they give examples where incomplete information increases social welfare.

Song and van der Schaar [49] extend previous work by considering the impact of foresight on network formation. Most network models assume that agents make myopic decisions. As a result, agents link with other agents if the payoff, or their beliefs about the payoff under incomplete information, of forming each link is positive. However, this paper extends the previous models by allowing agents to be foresighted and consider future payoffs in making links. The model is a discrete-time model like that in Song and van der Schaar [48]. It considers the limit as agents become very patient. They derive a "folk theorem" like result that shows that many networks are now possible. The idea is that when agents are very patient, it is possible to enforce a wide range of equilibrium strategies through imposing sufficiently long punishments when agents deviate from equilibrium play. Given that agents are very patient, they value the future greatly and thus will not wish to be punished. This paper shows that the range of possible networks can thus be even larger when foresight is introduced.

Zhang and van der Schaar [50] consider learning that occurs gradually, in contrast to Song and van der Schaar [48] and Song and van der Schaar [49] in which the learning is instant. Learning happens in continuous time, and the rate at which learning occurs will impact the resulting social welfare. In their model, agents who provide high benefits develop high reputations and remain in the network while agents who provide low benefits drop in reputation and become ostracized. The information to which agents have access and the speed at which they learn and act both have a tremendous impact on the resulting network dynamics. Importantly, the authors show that learning can have a negative impact on social welfare because it causes agents to become ostracized from the network more quickly. When an agent is ostracized from the network due to the learning process, although the ostracized agent may be of marginal quality it is still receiving a large benefit from the rest of the network. Thus, having the ostracized agent leave the network reduces social welfare on average. Faster learning may be undesirable as it leads to faster ostracization times. Due to the potential negative consequences of ostracism, it may be necessary to place agents with lower initial reputations at less central positions within the network. For instance, core-periphery networks with high-reputation agents in the core and low-reputation agents in the periphery can be optimal.

Zhang and van der Schaar [51] consider an alternative learning model in which agents exhibit homophilic preferences. Homophily is the preference to link with others who are similar to themselves. Agents have a limited capacity for links and thus maintain links with others learned to be similar to themselves and cut links to those learned to be dissimilar to themselves. Due to noise in the learning process, the agents may not find out which other agents are the closest in type, and thus the incomplete information network can exhibit vast differences from the complete information network.

The paper shows that higher levels of homophily decrease the (average) number of links that agents form. Mutually beneficial links may be dropped before learning is completed, thereby resulting in sparser networks and less clustering than under complete information. Homophily also exhibits an interesting interaction with the presence of incomplete information: initially, greater levels of homophily increase the difference between the complete and incomplete information networks, but sufficiently high levels of homophily eventually decrease the difference. Complete and incomplete information networks differ most when the degree of homophily is intermediate. The paper also extends the analysis to multiple stages of life/technology. Under such circumstances, the paper shows that the effects of incomplete information are large initially but fade somewhat over time.

26.5 CONCLUSION

In the modern world, applications that require multiple decentralized agents to act in harmony are ubiquitous. It is thus essential to understand what types of interactions enable them to cooperate and exchange information in order to reach their goals. This chapter presents learning strategies and algorithms for multiagent systems. It investigates how cooperative and strategic learning will lead to different types of networks between the agents, and how efficient these networks are in making the agents achieve their goals.

When the agents are cooperative, they are able to acquire knowledge about their environment by helping the other agents receive higher rewards, which also allows them to select better actions in the future. In addition, they are able to receive higher rewards by using the actions of the other agents. However, the willingness to cooperate depends on the cost of cooperation. If this cost is too high, agents will prefer to rely only on their own actions. It is also possible that agents' actions and rewards are correlated with each other. In such a case, the agents are required to follow distributed coordination strategies that will let them learn joint actions that maximize a global reward. With proper coordination, it is possible for agents to settle down to a joint action that maximizes the global reward by only observing noisy versions of it even without knowing the actions taken by the other agents.

On the other hand, strategic learning leads to many different kinds of networks between the agents, some of which can be efficient while the others can be highly inefficient. All the methods covered in this chapter shed light into how real-world networks are formed and how agents can reach their goals in these networks.

REFERENCES

[1] Schranz M, Dustdar S, Platzer C. Building an integrated pan-European news distribution network. In: Collaborative networks and their breeding environments, vol. 186; 2005. p. 587–96.

[2] Boutet A, Kloudas K, Kermarrec AM. FStream: a decentralized and social music streamer. In: Networked systems. Springer; 2013. p. 253–7.

[3] Mohan R, Smith JR, Li CS. Adapting multimedia internet content for universal access. IEEE Trans Multimedia 1999;1(1):104–14.

[4] Tekin C, van der Schaar M. Contextual online learning for multimedia content aggregation. IEEE Trans Multimedia 2015;17(4):549–61.

[5] Elmore JG, Miglioretti DL, Reisch LM, Barton MB, Kreuter W, Christiansen CL, et al. Screening mammograms by community radiologists: variability in false-positive rates. J Natl Cancer Inst 2002;94(18):1373–80.

[6] Trottier D. Social media as surveillance. Farnham: Ashgate; 2012.

[7] Chou WYS, Hunt YM, Beckjord EB, Moser RP, Hesse BW. Social media use in the United States: implications for health communication. J Med Internet Res 2009;11(4).

[8] Sakaki T, Okazaki M, Matsuo Y. Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th international conference on world wide web. ACM; 2010. p. 851–60.

[9] Choi H, Varian H. Predicting the present with Google Trends. Econ Rec 2012;88(s1):2–9.

[10] Liu HH, Wang Y, Yang YR, Wang H, Tian C. Optimizing cost and performance for content multihoming. In: Proceeding of the ACM SIGCOMM 2012 conference on applications, technologies, architectures, and protocols for computer communication. ACM; 2012. p. 371–82.

[11] Szabo G, Huberman BA. Predicting the popularity of online content. Commun ACM 2010;53(8):80–8.

[12] Cha M, Kwak H, Rodriguez P, Ahn YY, Moon S. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In: Proceedings of the 7th ACM SIGCOMM conference on internet measurement. ACM; 2007. p. 1–14.

[13] Wu T, Timmers M, De Vleeschauwer D, Van Leekwijck W. On the use of reservoir computing in popularity prediction. In: Proceedings of the 2nd international conference on evolving internet. IEEE; 2010. p. 19–24.

[14] Pinto H, Almeida JM, Gonçalves MA. Using early view patterns to predict the popularity of YouTube videos. In: Proceedings of the 6th ACM international conference on web search and data mining. ACM; 2013. p. 365–74.

[15] Li H, Ma X, Wang F, Liu J, Xu K. On popularity prediction of videos shared in online social networks. In: Proceedings of the 22nd ACM international conference on information & knowledge management. ACM; 2013. p. 169–78.

[16] Zhang L, Wang F, Liu J. Understand instant video clip sharing on mobile platforms: Twitter's Vine as a case study. In: Proceedings of the network and operating system support on digital audio and video workshop. ACM; 2014. p. 85.

[17] Galuba W, Aberer K, Chakraborty D, Despotovic Z, Kellerer W. Outtweeting the Twitterers-predicting information cascades in microblogs. WOSN 2010;10:3–11.

[18] Hong L, Dan O, Davison BD. Predicting popular messages in Twitter. In: Proceedings of the 20th international conference on companion on world wide web. ACM; 2011. p. 57–8.

[19] Lerman K, Hogg T. Using a model of social dynamics to predict popularity of news. In: Proceedings of the 19th international conference on world wide web. ACM; 2010. p. 621–30.

[20] Tekin C, van der Schaar M. Distributed online learning via cooperative contextual bandits. IEEE Trans Signal Process 2015;63(14):3700–14.

[21] Slivkins A. Contextual bandits with similarity information. J Mach Learn Res 2014;15(1):2533–68.

[22] Lu T, Pál D, Pál M. Contextual multi-armed bandits. In: Proceedings of the AISTATS; 2010. p. 485–92.

[23] Choi SS, Cha SH, Tappert CC. A survey of binary similarity and distance measures. J Syst Cybern Inf 2010;8(1):43–8.

[24] Cha SH. Comprehensive survey on distance/similarity measures between probability density functions. City 2007;1(2):1.

[25] Drugan MM, Nowe A. Designing multi-objective multi-armed bandits algorithms: a study. In: Proceedings of the international joint conference on neural networks (IJCNN); 2013. p. 1–8.

[26] Turgay E, Oner D, Tekin C. Multi-objective contextual bandit problem with similarity information. In: Proceedings of the AISTATS; 2018. p. 1673–81.
[27] Tekin C, Turgay E. Multi-objective contextual bandits with a dominant objective. In: Proceedings of the 27th IEEE international workshop on machine learning for signal processing; 2017.
[28] Cesa-Bianchi N, Lugosi G. Combinatorial bandits. J Comput Syst Sci 2012;78(5):1404–22.
[29] Auer P, Cesa-Bianchi N, Fischer P. Finite-time analysis of the multiarmed bandit problem. Mach Learn 2002;47(2-3):235–56.
[30] Anantharam V, Varaiya P, Walrand J. Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays—Part I: IID rewards. IEEE Trans Autom Control 1987;32(11):968–76.
[31] Anandkumar A, Michael N, Tang AK, Swami A. Distributed algorithms for learning and cognitive medium access with logarithmic regret. IEEE J Sel Areas Commun 2011;29(4):731–45.
[32] Gai Y, Krishnamachari B, Jain R. Combinatorial network optimization with unknown variables: multi-armed bandits with linear rewards and individual observations. IEEE/ACM Trans Netw 2012;20(5):1466–78.
[33] Chen W, Wang Y, Yuan Y. Combinatorial multi-armed bandit: general framework and applications. In: Proceedings of the international conference on machine learning (ICML); 2013. p. 151–9.
[34] Rusmevichientong P, Tsitsiklis JN. Linearly parameterized bandits. Math Oper Res 2010;35(2):395–411.
[35] Auer P. Using confidence bounds for exploitation-exploration trade-offs. J Mach Learn Res 2002;3:397–422.
[36] Dani V, Hayes TP, Kakade SM. Stochastic linear optimization under bandit feedback. In: COLT; 2008. p. 355–66.
[37] Szorenyi B, Busa-Fekete R, Hegedus I, Ormándi R, Jelasity M, Kégl B. Gossip-based distributed stochastic bandit algorithms. In: Proceedings of the international conference on machine learning (ICML); 2013. p. 19–27.
[38] Xu J, Tekin C, Zhang S, van der Schaar M. Distributed multi-agent online learning based on global feedback. IEEE Trans Signal Process 2015;63(9):2225–38.
[39] Foo B, van der Schaar M. A distributed approach for optimizing cascaded classifier topologies in real-time stream mining systems. IEEE Trans Image Process 2010;19(11):3035–48.
[40] Foo B, van der Schaar M. A rules-based approach for configuring chains of classifiers in real-time stream mining systems. EURASIP J Adv Signal Process 2009;2009:40.
[41] Golub B, Jackson MO. Naive learning in social networks and the wisdom of crowds. Am Econ J: Microecon 2010;2(1):112–49.
[42] Acemoglu D, Dahleh MA, Lobel I, Ozdaglar A. Bayesian learning in social networks. Rev Econ Stud 2011;78(4):1201–36.
[43] Acemoglu D, Bimpikis K, Ozdaglar A. Dynamics of information exchange in endogenous social networks. Theor Econ 2014;9(1):41–97.
[44] Song Y. Social learning with endogenous network formation; 2015. arXiv preprint arXiv:150405222.
[45] Jackson MO, Wolinsky A. A strategic model of social and economic networks. J Econ Theory 1996;71(1):44–74.
[46] Bala V, Goyal S. A noncooperative model of network formation. Econometrica 2000;68(5):1181–229.
[47] Jackson MO, Watts A. The evolution of social and economic networks. J Econ Theory 2002;106(2):265–95.
[48] Song Y, van der Schaar M. Dynamic network formation with incomplete information. Econ Theory 2015;59(2):301–31.
[49] Song Y, van der Schaar M. Dynamic network formation with foresighted agents; 2015. arXiv preprint arXiv:150900126.
[50] Zhang S, van der Schaar M. Reputational dynamics in financial networks during a crisis. Office of Financial Research, US Department of the Treasury, Working Paper: No. 18-03; 2018.
[51] Zhang S, van der Schaar M. From acquaintances to friends: homophily and learning in networks. IEEE J Sel Areas Commun 2017;35(3):680–90.