Communication-Free Distributed Coverage for Networked Systems

A. Yasin Yazıcıoğlu, Magnus Egerstedt, and Jeff S. Shamma

Abstract—In this paper, we present a communication-free algorithm for distributed coverage of an arbitrary network by a group of mobile agents with local sensing capabilities. The network is represented as a graph, and the agents are arbitrarily deployed on some nodes of the graph. Any node of the graph is covered if it is within the sensing range of at least one agent. The agents are mobile devices that aim to explore the graph and to optimize their locations in a decentralized fashion by relying only on their sensory inputs. We formulate this problem in a game theoretic setting and propose a communication-free learning algorithm for maximizing the coverage.

I. INTRODUCTION

In many networked systems, a typical task is to provide some service such as security or maintenance via some agents with limited capabilities (e.g., [1], [2], [3]). One way of achieving this task is to solve a locational optimization problem (e.g., [4], [5], [6], [7], [8]) and let each agent serve some part of the network around its assigned position. In the absence of a centralized mechanism, the agents are faced with a distributed coverage control problem, where their objective is to optimize their locations by following some decentralized controllers.

Distributed coverage control is widely studied on continuous domains (e.g., [9]-[10]). One possible approach is to employ potential fields to drive each agent away from the nearby agents and obstacles (e.g., [9], [11]). Alternatively, a prevailing approach introduced in [12] is to model the underlying locational optimization problem as a continuous $p$-median problem and to employ Lloyd’s algorithm [13]. As such, the agents are driven onto a local optimum, i.e., a centroidal Voronoi configuration, where each point in the space is assigned to the nearest agent, and each agent is located at the center of mass of its own region. Later on, this method was extended for agents with distance-limited sensing and communications (e.g., [14]) and limited power (e.g., [15]), as well as for heterogeneous agents covering non-convex regions (e.g., [16]).

Also, the requirement of sensing density functions was relaxed by incorporating methods from adaptive control and learning (e.g., [10]).

In some studies, distributed coverage control was studied on discrete spaces represented as graphs (e.g., [17], [18], [19], [20]). One possible approach is to achieve a centroidal Voronoi partition of the graph via pairwise gossip algorithms (e.g., [17]) or via asynchronous greedy updates (e.g., [18]).

Alternatively, distributed coverage control on discrete spaces can be studied in a game theoretic framework (e.g., [19], [20]). Game theoretic methods have been used to solve many cooperative control problems such as vehicle-target assignment (e.g., [21]), dynamic vehicle routing (e.g., [22]), cooperative communication (e.g., [23]), and coverage optimization (e.g., [19], [20]). In [19], sensors with variable footprints achieve power-aware optimal coverage on a discretized space. In [20], a group of heterogeneous mobile agents are driven on a graph to maximize the number of covered nodes.

In this paper, we study a distributed coverage control problem on graphs in a game theoretic setting. In this problem, mobile agents are arbitrarily deployed on an unknown graph. Each agent is assumed to sense the local graph structure and the presence of other agents (if any) within its sensing range. Any node of the graph is covered if it is within the sensing range of at least one agent. The objective of the agents is to maximize the number of covered nodes by optimizing their locations on the graph. We present a game theoretic formulation for this coverage control problem. We particularly focus on a communication-free setting, where each agent should be driven via only its sensory inputs. In that case, the agents do not observe their exact utilities in the corresponding game. Accordingly, we propose a learning algorithm for driving the agent positions based on some estimated utilities. Using the proposed method, the agents maintain optimal coverage with an arbitrarily high probability as time goes to infinity.

The organization of this paper is as follows: Section II presents the distributed graph coverage problem. Section III sets up the game-theoretic formulation of the problem. Section IV presents a solution that requires some explicit communications among the agents. The proposed communication-free solution is presented in Section V. Some simulation results for the proposed method are presented in Section VI. Finally, Section VII concludes the paper.

II. DISTRIBUTED GRAPH COVERAGE

In this section, we present the distributed graph coverage (DGC) problem, where the goal is to maximize the number of covered nodes by driving the agents with limited sensing and mobility capabilities to optimal locations on a graph. First, some graph theory preliminaries are presented.

A. Graph Theory Concepts

An undirected graph $G = (V, E)$ consists of a node set $V$ and an edge set $E \subseteq V \times V$. For an undirected graph, the edge set consists of unordered node pairs $(v, v')$ denoting that the nodes $v$ and $v'$ are adjacent.

A path is a sequence of nodes such that each node is adjacent to the preceding node in the sequence. For any two
nodes \( v \) and \( v' \), the \textit{distance} between the nodes \( d(v, v') \) is the number of edges in a shortest path between \( v \) and \( v' \). A graph is \textit{connected} if the distance between any pair of nodes is finite.

The set of nodes containing a node \( v \) and all the nodes adjacent to \( v \) is called the \textit{(closed) neighborhood} of \( v \), and it is denoted as \( N_v \). For any \( \delta \geq 0 \), the \( \delta \)-neighborhood of \( v \), \( N_v^{\delta} \), is the set of nodes that are at most \( \delta \) away from \( v \), i.e.

\[
N_v^{\delta} = \left\{ v' \in V \mid d(v, v') \leq \delta \right\}.
\]

For any \( G = (V, E) \), an induced subgraph, \( G[V_s] \), consists of the vertices, \( V_s \subseteq V \), and the edges whose endpoints are both in \( V_s \).

### B. Problem Formulation

Consider a connected undirected graph, \( G = (V, E) \), and let \( I = \{1, 2, \ldots, m\} \) denote a set of \( m \) mobile agents arbitrarily deployed on some nodes of the graph. Let each agent have a sensing range, \( \delta \). We assume that each agent, \( i \), can sense the subgraph induced by the nodes in \( N_{v_i}^{\delta} \), and the presence of other agents (if any) within its \( \delta \)-neighborhood. As such, each \( i \in I \) located at \( v_i \in V \) covers all the nodes in \( N_{v_i}^{\delta} \). Any node of the graph is covered if it is included in the \( \delta \)-neighborhood of at least one agent, and the set of covered nodes, \( V_c \subseteq V \), is given as

\[
V_c(v_1, \ldots, v_m) = \bigcup_{i=1}^{m} N_{v_i}^{\delta}.
\]

The objective in the distributed graph coverage (DGC) problem is to have the agents update their positions over time in a distributed manner to maximize the number of covered nodes, i.e.

\[
|V_c(v_1(t), \ldots, v_m(t))|, \quad \text{(3)}
\]

where each \( v_i(t) \in V \) is the position of agent \( i \) at time \( t \).

In order to achieve optimal coverage in a distributed fashion, the agents need some local rules to follow. In general, a rule is considered to be local if its execution by an agent requires only some information available within a small distance from the agent. In this paper, we consider a discrete time dynamics and we assume that each agent can either maintain its position or move to an adjacent node in the next time step, i.e.

\[
d(v_i(t+1), v_i(t)) \leq 1, \forall i \in \{1, 2, \ldots, m\}. \text{ (4)}
\]

In this setting, multiple agents can be located on the same node, and the only constraint on the agent movements is the locality constraint in (4).

### C. Solution Approach

In the DGC problem, a group of mobile agents explore an unknown graph and aim to cover as many nodes as possible. As such, the underlying locational optimization problem is similar to the maximum coverage problem (e.g., [5], [6]). Such NP-hard problems are typically tackled by finding sufficiently good approximate solutions through fast algorithms (e.g. [24], [25], [26]). Similarly, in many distributed coverage control studies, a locational objective function is optimized by the agents aiming for the best local improvements (e.g., [12]-[18]). Such a distributed greedy approach can be employed to solve the DGC problem. Accordingly, the agents may move locally on the graph to maximally improve their local coverage. In that case, the resulting performance would significantly depend on the graph structure and the initial configuration. This method may rapidly lead to a reasonable approximate solution if the agents start with a sufficiently good initial coverage or if the interaction graph satisfies some structural properties. However, it may also lead to arbitrarily poor configurations for arbitrary graphs and initial conditions. For instance, consider the scenario in Fig. 1, where 2 agents with sensing ranges \( \delta = 1 \) can achieve a globally optimal configuration in 2 time steps. In this example, the initial configuration would be stationary under a greedy approach since none of the agents can improve the coverage by moving to a neighboring node. Note that the performance in Fig. 1a would be arbitrarily poor for any arbitrarily large graph obtained by adding more leaf nodes attached to the unoccupied hub.

![Fig. 1. A possible trajectory to a globally optimal configuration for two agents on a small graph. The agents have cover ranges \( \delta = 1 \), and they are initially located as in (a). The number of covered nodes (shown as non-white) is reduced in the intermediate step illustrated in (b) to reach the global optima shown in (c).](image)

In order to ensure efficient coverage for arbitrary graphs and initial configurations, a solution method should occasionally allow for graph exploration at the expense of a better coverage. In this work, we present such a solution by approaching the problem from a game theoretic perspective. In particular, we map the DGC problem to a game, and we design a learning algorithm for the agents to follow in updating their actions.

### III. GAME THEORETIC FORMULATION

In this section, a game-theoretic formulation of the DGC problem is presented. First, some game theory preliminaries are provided.

#### A. Game Theory Concepts

A finite \textit{strategic game} \( \Gamma = (I, A, U) \) consists of three components: (1) a set of \( m \) players (agents) \( I = \{1, 2, \ldots, m\} \), (2) an \( m \)-dimensional action space \( A = A_1 \times A_2 \times \ldots \times A_m \), where each \( A_i \) is the action set of player \( i \), and (3) a set of \textit{utility functions} \( U = \{U_1, U_2, \ldots, U_m\} \), where each \( U_i : A \rightarrow \mathbb{R} \) is a mapping from the action space to real numbers.

For any action profile \( a \in A \), let \( a_{-i} \) denote the actions of players other than \( i \). Using this notation, an action profile \( a \)
A class of games that is widely utilized in cooperative control problems is the potential games. A game is called a potential game if there exists a potential function, $\phi : A \rightarrow \mathbb{R}$, such that the change of a player’s utility resulting from an unilateral deviation from an action profile equals the resulting change in $\phi$. More precisely, for each player $i$, for every action profile $a_i, a_{-i} \in A_i$, and for all $a_{-i} \in A_{-i}$,

$$U_i(a_i', a_{-i}) - U_i(a_i, a_{-i}) = \phi(a_i', a_{-i}) - \phi(a_i, a_{-i}).$$

(6)

When a cooperative control problem is mapped to a potential game, the game is designed such that its potential function captures the global objective (e.g., [27]).

B. DGC Game

In order to formulate the DGC problem in a game theoretic setting, we design a corresponding game, $\Gamma_{DGC}$, by defining the action space and the utility functions. More specifically, we design a potential game such that its potential function, $\phi(a)$, captures the global objective of the DGC problem, i.e.

$$\phi(a) = |V_i(a)|.$$  

(7)

In the DGC problem, the coverage provided by each agent is determined by the position of the agent. Hence, the action of an agent can be defined as its position on the graph. Accordingly, each action set is equal to the node set of $\mathcal{G} = (V, E)$, i.e.

$$A_i = V, \quad \forall i \in I.$$  

(8)

Then, the utilities should be designed such that $\phi(a)$ in (7) is indeed the potential function for the resulting game. To this end, we design the agent utilities as

$$U_i(a) = \sum_{v \in N_a^i} u_i(v, a_{-i}),$$  

(9)

where, for every $v \in N_a^i$, $u_i(v, a_{-i})$ is the partial utility agent $i$ gathers by covering node $v$, and it is defined as

$$u_i(v, a_{-i}) = \begin{cases} 
1 & \text{if } d(v, a_j) > \delta \quad \forall j \neq i, \\
0 & \text{otherwise}.
\end{cases}$$

(10)

In the resulting game, each agent gathers a utility equal to the number of nodes that are covered only by itself. Note that this utility is equal to the marginal contribution of the corresponding agent to the number of covered nodes.

Lemma 3.1. The utilities in (9) lead to a potential game $\Gamma_{DGC} = (P, A, U)$ with the potential function given in (7).

Proof. Let $a_i = v_j$ and $a'_i = v'_j$ be two possible actions for agent $i$, and let $a_{-i}$ denote the actions of other agents. Due to (2) and (7),

$$\phi(a) = \sum_{i \in I} |N_a^i|$$

(11)

Using (9), for any agent $i$, (11) can be expanded as

$$\phi(a) = |N_a^i| \setminus \bigcup_{j \neq i} N_a^j| + | \bigcup_{j \neq i} N_a^j| = U_i(a_i, a_{-i}) + \bigcup_{j \neq i} N_a^j|.$$  

(12)

Using (12) for any pair of actions $a_i$ and $a_i'$,

$$\phi(a_i, a_{-i}) - \phi(a_i', a_{-i}) = U_i(a_i', a_{-i}) - U_i(a_i, a_{-i}).$$

(13)

C. Learning

In game theoretic learning, starting from an arbitrary initial configuration, the agents repetitively play a game. At each step $t \in \{0, 1, 2, \ldots\}$, each agent $i \in I$ plays an action $a_i(t)$ and receives some utility $U_i(a(t))$. In this setting, the agents update their actions in accordance with some learning algorithms. For the DGC problem, the learning process is desired to drive the agent positions to the set of configurations that maximize the number of covered nodes.

For potential games, there are many learning algorithms that provide convergence to a Nash equilibrium (e.g., [21], [27] and the references therein). While any potential maximizer is a Nash equilibrium, a potential game may also have some suboptimal Nash equilibria. In that case, a learning algorithm such as log-linear learning (LLL) can be used to drive the agents to the configurations that maximize the potential function $\phi(a)$ [28]. Essentially, LLL is a noisy best-response algorithm, and it induces a Markov chain over the action space with a unique limiting distribution, $\mu^*$, where $\epsilon$ denotes the noise parameter. As the noise parameter, $\epsilon$, goes down to zero, the limiting distribution, $\mu^*$, has an arbitrarily large part of its mass over the set of potential maximizers [28]. However, LLL assumes that at any round each player $i$ has access to all the actions in its action set $A_i$. In general, LLL may not provide potential maximization when the system evolves over constrained action sets, i.e. when each agent $i$ is allowed to choose its next action from only a subset of actions. Note that this is indeed the case for the DGC problem, and each agent has to pick its the next action from the closed neighborhood of its current action $a_i$,

$$A_i^*(a_i) = N_a^i, \forall i \in I.$$  

(14)

The issue of constrained action sets was addressed in [29], and a variant learning algorithm called binary log-linear learning (BLLL) was presented for such cases.

In learning algorithms, typically each agent is assumed to measure its current utility. For instance, in order to execute LLL or BLLL, the agents need to measure their utilities resulting from their current actions as well as the hypothetical utilities they may gather by unilaterally switching to some other actions. Alternatively, a payoff-based implementation may be utilized to avoid the necessity to compute the hypothetical utilities [29]. Note that, for $\Gamma_{DGC}$, even the computation of the current utility requires some explicit communications since the agents with overlapping coverage are not necessarily within the sensing range of each other. In general, such agents can be up to $2\delta$ apart on the graph.
D. Stochastic Stability Concepts

Stochastic stability was introduced in game theory to explain how some equilibria may arise much more frequently than the others in the long run as the strategic decision makers update their actions in a stochastic setting with noise [28], [30], [31]. The concept of stochastic stability will be extensively used in the remainder of this paper. Hence, we provide some preliminaries prior to our derivations, and further details about the subject can be found in [28], [30], [31] and the references therein.

Definition (Regular Perturbed Markov Chain): Let \( P \) be the transition matrix of a discrete-time Markov chain over a finite state space \( \mathcal{X} \). A perturbed Markov chain with the noise parameter \( \epsilon \) is called a regular perturbed Markov chain if

1. \( P_\epsilon \) is aperiodic and irreducible for \( \epsilon > 0 \),
2. \( \lim_{\epsilon \to 0} P_\epsilon = P \).
3. For any \( x, x^+ \in \mathcal{X} \) if \( P_\epsilon(x, x^+) > 0 \), then there exists \( R(x, x^+) \geq 0 \) such that

\[
0 < \lim_{\epsilon \to 0^+} \frac{P_\epsilon(x, x^+)}{\epsilon R(x, x^+)} < \infty,
\]

where \( R(x, x^+) \) is called the resistance of the transition from \( x \) to \( x^+ \).

Any regular perturbed Markov chain, \( P_\epsilon \), has a unique limiting distribution, \( \mu^*_\epsilon \), since it is aperiodic and irreducible. In a stochastic stability analysis, the objective is to investigate the behavior of \( \mu^*_\epsilon \) as the noise parameter \( \epsilon \) goes down to zero. In a nutshell, a state is stochastically stable if the corresponding entry of the limiting distribution does not approach zero as the noise in the process becomes arbitrarily small.

Definition (Stochastically Stable State): Let \( P_\epsilon \) denote a regular perturbed Markov chain over a state space, \( \mathcal{X} \). Any state, \( x \in \mathcal{X} \), is stochastically stable if

\[
\lim_{\epsilon \to 0^+} \mu^*_\epsilon(x) > 0.
\]

The stochastically stable states of any regular perturbed Markov chain, \( P_\epsilon \), can be characterized through a resistance tree analysis. For any state \( x \in \mathcal{X} \) of a regular perturbed Markov chain, a spanning tree rooted at \( x \) can be constructed as a directed graph \( T_x \), where the nodes correspond to states, the edges correspond to feasible state transitions, and there is a unique directed path on \( T_x \) from any state \( x' \neq x \) to \( x \). The resistance of such a tree, \( R(T_x) \), is defined as the sum of the resistances of its edges, where the resistance of each edge is given as in (15). A spanning tree \( T^*_x \) is called a minimum resistance tree if \( R(T^*_x) \leq R(T_x) \) for any \( T_x \), i.e., any spanning tree rooted at \( x \) has at least as much resistance as \( T^*_x \). The stochastic potential of a state, \( x \), is defined as the total resistance of its minimum resistance tree, \( R(T^*_x) \). The following result characterizes the stochastically stable states through their stochastic potentials.

Lemma 3.2. [30] Let \( P_\epsilon \) be a regular perturbed Markov chain. A state \( x \in \mathcal{X} \) is stochastically stable if and only if \( x \) is a recurrent state of the unperturbed chain, \( P_0 \), with the minimum stochastic potential.

IV. COVERAGE MAXIMIZATION

In this section, we will briefly show that if all the agents follow BLLL in a repetitive play of \( \Gamma_{DGC} \), then the stochastically stable states are the coverage maximizers. A more detailed presentation of this approach can be found in [20]. As stated earlier, this solution requires some local communications among the agents. In the next section, we will present a novel learning algorithm that can achieve the same limiting behavior as this method without any explicit communications.

Algorithm 1: Binary Log-linear Learning ([29])

1. initialization: \( \epsilon \in \mathbb{R}^+ \) small, \( a \in A \) arbitrary
2. repeat
3. Pick a random \( i \in I \), and a random \( a'_i \in A_i^r(a_i) \).
4. Compute \( \alpha = \epsilon^{-U_i(a_i(t))} \), \( \beta = \epsilon^{-U_i(a'_i,a_{-i}(t))} \).
5. With probability \( \frac{\beta}{\alpha+\beta} \), set \( a_i = a'_i \).
6. end repeat

In BLLL, a single agent is randomly chosen at each time step. The selection of a single agent at each time step can be achieved (with a very high probability) without a centralized coordination by using methods such as the asynchronous time model proposed in [32]. The selected agent, assuming that all the other agents are stationary, updates its action depending on its current utility and the hypothetical utility it would receive by playing a random action in its constrained action set. This is illustrated in Fig. 2.

![Algorithm 1: Binary Log-linear Learning](image)

Fig. 2. An illustration of the BLLL algorithm, where two agents with \( \delta = 1 \) are located as in (a) and Agent 1 is updating its action. Agent 1 randomly picks a candidate action, \( a'_1 \in A_1^r(a_1) \), as in (b). Its next action is picked from \( \{a_1, a'_1\} \) with probabilities depending on the corresponding utilities. For the configuration in (b), the tiled node is not providing any utility to either of the agents since it is covered by both of them.

In [29], it was shown that BLLL can be used to achieve potential maximization if the constrained action sets satisfy Properties 1 and 2 provided below.

Property 1 (Reachability) For any agent \( i \in I \) and any action pair \( a_i^0, a_i^r \in A_i \), there exists a sequence of actions \( \{a_i^0, a_i^1, \ldots, a_i^k\} \) such that \( a_i^r \in A_i^r(a_i^{r-1}) \) for all \( r \in \{1, 2, \ldots, k\} \).

Property 2 (Reversibility) For any agent \( i \in I \) and any action pair \( a_i, a'_i \in A_i \),

\[
a'_i \in A_i^r(a_i) \iff a_i \in A_i^r(a'_i).
\]
Theorem 4.1. [29] Consider any finite potential game and constrained action sets satisfying Properties 1 and 2. If all players adhere to BLLL, then the stochastically stable states are the set of potential maximizers.

In light of Theorem 4.1, the agents can maximize the coverage by following the BLL algorithm in a repetitive play of $\Gamma_{DGC}$, if the constrained action sets given in (14) satisfy Properties 1 and 2. Lemma 4.2 shows that the constrained action sets indeed satisfy these properties if the graph to be covered is connected.

Lemma 4.2. The constrained action sets in (14) satisfy Properties 1 and 2 if the graph $G = (V,E)$ is connected.

Proof. If the graph is connected, then there exists a finite-length path $\{v^0, \ldots, v^k\}$ between any pair of nodes $v^0, v^k \in V$, and Property 1 is satisfied. Furthermore, for undirected graphs, $d(v, v') = d(v', v)$. Hence, Property 2 is also satisfied.

Theorem 4.3. Let $G = (V,E)$ be connected graph, and let all agents follow BLL in a repetitive play of $\Gamma_{DGC}$ with the constrained action sets in (14). Then the stochastically stable states are the maximizers of $|V_c(a_i)|$.

Proof. If $G = (V,E)$ is connected, then the constrained action sets in (14) satisfy Properties 1 and 2 due to Lemma 4.2. Hence, in light of Theorem 4.1, if all agents follow BLLL in a repetitive play of $\Gamma_{DGC}$, the stochastically stable states are the potential maximizers. Due to (7), those are the configurations maximizing the number of covered nodes, $|V_c(a_i)|$.

V. COMMUNICATION-FREE COVERAGE MAXIMIZATION

In the DGC problem, the sensory inputs do not reveal which of the nodes within the sensing range of an agent is covered also by some other agents. However, each agent can sense if any other agent is also covering its current position as illustrated in Fig. 3. Hence, each agent $i$ observes the partial utility, $u_i(a_i, a_{-i})$ in (10), via its sensory input.

Since the exact utilities in $\Gamma_{DGC}$ are not measurable without explicit communications, the agents need to update their actions based on some estimated utilities. Assuming that the nearby agents will remain stationary for a sufficient amount of time, each agent $i$ can construct an estimated utility by visiting each $v \in N^\delta_{a_i}$, and combining the sampled $u_i(v, a_{-i})$.

Note that the resulting estimation will not necessarily be equal to the actual utility since multiple agents may be moving simultaneously as illustrated in Fig. 4. However, if the probability of having simultaneously moving agents is sufficiently small, then false estimations will be sufficiently rare for the agents to achieve the desired limiting behavior by following a noisy best-response based on the estimated utilities. The proposed communication-free algorithm is based on this approach, and it achieves the desired limiting behavior by making the erroneous utility estimations sufficiently rare without requiring any scheduling or coordination among the agents.

In the remainder of this section, we present the proposed communication-free coverage maximization algorithm (CFCM) and an analysis of the corresponding dynamics.

A. CFCM Algorithm

The proposed algorithm has two parameters to be set. The first parameter, $\epsilon \in \mathbb{R}^+$, is the noise in the agent decisions when choosing between the candidate actions based on the corresponding estimated utilities. The second parameter, $r \in \mathbb{R}^+$, sets the likelihood of each agent to update its action. As it will be shown later in this section, the desired global behavior emerges when $r$ is sufficiently large and $\epsilon$ is small.

In CFCM, each agent $i$ is either stationary or experimenting. Each stationary agent repeats its current action in the next time step with a high probability, $1 - e^r$, or starts an experiment with probability $e^r$. An experiment involves comparing its current action, $a_i^t$, to an alternative randomly picked from its constrained action set, $a_i^t \in A_i^c(a_i^t)$, where $A_i^c(a_i^t)$ is the local neighborhood of $a_i^t$ as given in (14). In this aspect, the agent behavior is similar to the payoff-based BLL in [29]. However, since the agents receive only some partial utilities, $u_i(a_i, a_{-i})$, an experiment consists of visiting all the nodes in $N^\delta_{a_i^t} \cup N^\delta_{a_i^t, i}$ to see which of those nodes are also covered by some other agents. We refer to the corresponding path to be traversed as an experiment path between $a_i^t$ and $a_i^t$.

Definition (Experiment Path): Let $\delta$ be the sensing range of the agents. For any $a_i^t$ and $a_i^t \in A_i^c(a_i^t)$, a finite path, $\{a_i^t, \ldots, a_i^t\}$, is an experiment path if it traverses $N^\delta_{a_i^t} \cup N^\delta_{a_i^t, i}$.
For any \( a_1^i \) and \( a_2^i \in A_c^i(a_1^i) \), an experiment path can be obtained locally by utilizing methods such as depth-first search or breadth-first search (e.g., [33]). In the CFCM algorithm, an experiment path between \( a_1^i \) and \( a_2^i \) is denoted as \( E(a_1^i, a_2^i) \). During an experiment, the agent traverses its experiment path to construct the estimated utilities, \( \hat{U}_1^i \) and \( \hat{U}_2^i \), from the sampled partial utilities. For simplicity, a partial utility from a node is sampled only at the last visit to that node during the experiment. As such, if it is the agent’s last visit of the current position, \( a_i \), and the agent does not sense any other agent within \( \delta \), then the utility estimations corresponding to the candidate actions within \( \delta \) from \( a_i \) are incremented by 1. Once the experiment path is traversed, the agent randomly chooses between the two candidate actions based on the estimated utilities, \( \hat{U}_1^i \) and \( \hat{U}_2^i \). At the next time step, the agent becomes stationary at its chosen action until it starts a new experiment.

\[
[S_i(k_i)] \text{ denotes the } k_i^{th} \text{ element in } S_i, \quad \text{and } |S_i| \text{ denotes the length of } S_i. \text{ The CFCM algorithm is memoryless since the state of every agent in the next time step is independent of its past trajectory. As such, if all agents follow the CFCM algorithm, then a Markov chain is induced over the state space, } \mathcal{X}, \text{ where each } x \in \mathcal{X} \text{ is the global state obtained by concatenating the states of all agents, i.e.}
\]

\[
x = [x_1, x_2, \ldots, x_m].
\]

In the remainder of this section, the limiting behavior of the resulting Markov chain will be inspected through a stochastic stability analysis.

### B. Limiting Behavior

For any \( x \in \mathcal{X} \), the agents can be grouped into two distinct sets consisting of the stationary agents, \( I_s(x) \), and the experimenting agents, \( I_e(x) \), as

\[
I_s(x) = \{ i \in I \mid |S_i| = 1 \}, \quad I_e(x) = I \setminus I_s(x).
\]

Using these sets, for any feasible transition, \( x \rightarrow x^+ \), the agents can be grouped into 4 disjoint sets based on the transition of their individual states:

\[
I_{s_s}(x, x^+) = I_s(x) \cap I_s(x^+), \quad I_{s_e}(x, x^+) = I_s(x) \cap I_e(x^+), \quad I_{e_s}(x, x^+) = I_e(x) \cap I_s(x^+), \quad I_{e_e}(x, x^+) = I_e(x) \cap I_e(x^+),
\]

where \( I_{s_s}(x, x^+) \) are the agents that remain stationary, \( I_{s_e}(x, x^+) \) are the ones starting to experiment, \( I_{e_s}(x, x^+) \) are the experimenting agents that have not completed moving along their experiment paths, and \( I_{e_e}(x, x^+) \) are the agents that have completed traversing their experiment paths and choose between their candidate actions.

The agents in \( I_{e_s}(x, x^+) \) can be further partitioned as the ones choosing their first candidate action and the ones that choose their second candidate action, i.e.

\[
I_{e_s}^1(x, x^+) = \{ i \in I_{e_s}(x, x^+) \mid a_1^i = a_1^i \}, \quad I_{e_s}^2(x, x^+) = \{ i \in I_{e_s}(x, x^+) \mid a_2^i = a_2^i \}.
\]

Note that the agents in \( I_{e_s}(x, x^+) \) do not necessarily choose the action resulting in the higher estimated utility. For each \( i \in I, \) let \( \hat{U}_i^* = \max\{\hat{U}_1^i, \hat{U}_2^i\} \). Then, the amount of estimated utility that is denied in the transition \( x \rightarrow x^+ \) is given as

\[
\Delta_i(x_i, x_i^+) = \begin{cases} \hat{U}_i^1 - \hat{U}_i^1 & \text{if } i \in I_{e_s}(x, x^+), \\ \hat{U}_i^2 - \hat{U}_i^2 & \text{if } i \in I_{e_s}^2(x, x^+), \\ 0 & \text{otherwise.} \end{cases}
\]

Next, we show that the CFCM algorithm induces a regular perturbed Markov chain, where the resistance of any feasible transition depends on the estimated utilities denied by the agents becoming stationary and the number of agents starting new experiments.

Algorithm II: Communication-free Coverage Maximization (CFCM)

1: initialization: \( \epsilon \in \mathbb{R}^+ \) (small), \( r \in \mathbb{R}^+ \), \( a_i \in A_i \), arbitrary,
\[
S_i = \{a_i \}, \quad k_i = 1, \quad \hat{U}_1^i = \hat{U}_2^i = 0.
\]
2: \[ \text{repeat} \]
3: \[ \quad a_i = S_i(k_i), \quad a_1^i = S_i(1), \quad a_2^i = S_i(|S_i|). \]
4: \[ \text{if } (|S_i| = 1) \]
5: \[ \quad \text{Generate a random (uniform) } \gamma \in [0, 1]. \]
6: \[ \quad a_2^i \text{ is randomly (uniform) chosen over } A_c^i(a_1^i). \]
7: \[ S_i = E(a_1^i, a_2^i). \]
8: \[ \text{end if} \]
9: \[ \text{end if} \]
10: \[ \text{if } (k_i \geq k, \forall k \in \{k \mid S_i(k) = a_i\}) \]
11: \[ \quad \hat{U}_1^i = \hat{U}_1^i + u_i(a_1^i, a_1^i), \quad \text{if } a_i \in A_c^i(a_1^i). \]
12: \[ \quad \hat{U}_2^i = \hat{U}_2^i + u_i(a_2^i, a_1^i), \quad \text{if } a_i \in A_c^i(a_2^i). \]
13: \[ \text{end if} \]
14: \[ \text{if } (k_i = |S_i|) \]
15: \[ \quad \alpha = e^{-\theta}, \quad \beta = e^{-\delta}, \]
16: \[ \quad S_i = \begin{cases} \{a_1^i\} \quad \text{w.p. } \frac{\alpha}{\alpha + \beta}, \\ \{a_2^i\} \quad \text{otherwise.} \end{cases} \]
17: \[ \quad k_i = 1, \quad \hat{U}_1^i = \hat{U}_2^i = 0. \]
18: \[ \text{else} \]
19: \[ \quad k_i = k_i + 1. \]
20: \[ \text{end if} \]
21: \[ \text{end if} \]
22: \[ \text{end if} \]
23: \[ \text{end repeat} \]
Lemma 5.1. Let \( G = (V, E) \) be connected graph. If all agents employ the CFCM algorithm, then a regular perturbed Markov chain is induced over \( X \), and the resistance of any feasible transition, \( x \to x^+ \), is

\[
R(x, x^+) = r|I_{se}(x, x^+)| + \sum_{i \in I_{se}(x, x^+)} \Delta_i(x_i, x_i^+). \tag{29}
\]

Proof. Let \( P_e \) denote the transition matrix of the Markov chain induced by the CFCM algorithm. For \( \epsilon > 0 \), any all-stationary state can be reached from any other all-stationary state through a sequence of experiments, given \( G = (V, E) \) is connected. Furthermore, any state that is not all-stationary lies on a feasible path between two all-stationary states. Hence, \( P_e \) is irreducible. Furthermore, since the stationary agents remain stationary with probability \( 1 - \epsilon^r \), aperiodicity immediately follows from the resulting self-loops at all-stationary states.

The probability any feasible transition from \( x \to x^+ \), given in \( P_e \), is the joint probability of state transitions of individual agents. Note that for any agent, \( i \in I_{se}(x) \), the transition from \( x_i \) to \( x_i^+ \) does not have any randomness. Hence, the probability of transition from \( x \) to \( x^+ \) is

\[
P_e(x, x^+) = \Pr[I_{es}(x, x^+)] \Pr[I_{es}^1(x, x^+)] \Pr[I_{ss}(x, x^+)] \Pr[I_{se}(x, x^+)], \tag{30}
\]

where each term on the right side of (30) denote the joint probability of state transitions for the agents in the corresponding subset, and they are given as

\[
\Pr[I_{es}(x, x^+)] = \prod_{i \in I_{es}(x, x^+)} \frac{\epsilon^{-U_1^i}}{\epsilon^{-U_1^i} + \epsilon^{-U_2^i}}, \tag{31}
\]
\[
\Pr[I_{es}^1(x, x^+)] = \prod_{i \in I_{es}^1(x, x^+)} \frac{\epsilon^{-U_2^i}}{\epsilon^{-U_2^i} + \epsilon^{-U_1^i}}, \tag{32}
\]
\[
\Pr[I_{ss}(x, x^+)] = \prod_{i \in I_{ss}(x, x^+)} (1 - \epsilon^r), \tag{33}
\]
\[
\Pr[I_{se}(x, x^+)] = \prod_{i \in I_{se}(x, x^+)} \frac{\epsilon^r}{|A_i^1(a_i^1)|} \Pr[S_i^1; a_i^1, a_i^2], \tag{34}
\]

where \( \Pr[S_i^1; a_i^1, a_i^2] \) is the probability of having \( S_i^1 \) as the experiment path for an agent comparing \( a_i^1 \) and \( a_i^2 \). \( \Pr[S_i^1; a_i^1, a_i^2] \) depends on the function \( E(a_i^1, a_i^2) \), and it is independent of \( \epsilon \). Plugging (31)-(34) into (30), one can verify that the resistance \( R(x, x^+) \) given in (29) satisfies

\[
0 < \lim_{\epsilon \to 0^+} \frac{P_e(x, x^+)}{\epsilon R(x, x^+)} < \infty. \tag{35}
\]

Since the CFCM algorithm induces a regular perturbed Markov chain, the stochastically stable states are the recurrent states of the unperturbed chain with the minimum stochastic potential, as given in Lemma 3.2. Note that if \( \epsilon = 0 \), then no agent starts an experiment. In that case, the set of recurrent states, \( X^0_R \), consists of the all-stationary states. All the other states, where at least one agent is experimenting, form the set of transient states, \( X^0_T \), i.e.

\[
X^0_R = \{x \mid I_{se}(x) = I\}, \tag{36}
\]
\[
X^0_T = X \setminus X^0_R. \tag{37}
\]

The stochastic potentials of the states in \( X^0_R \) are determined by the resistances of the feasible transitions. Note that the parameter \( r \) in the CFCM algorithm has a direct influence on the resistances as given in (29). We will show that, for any connected graph \( G \), if \( r \) is sufficiently large, then the states in \( X^0_R \) with the minimum stochastic potential are the coverage maximizers. To provide a sufficient value of \( r \), first we relate the structure of the graph to the maximum amount of estimated utility that can be denied by an agent in any feasible transition under the CFCM algorithm.

Lemma 5.2. Let all agents follow the CFCM algorithm to cover a connected graph, \( G = (V, E) \), and let \( \nu(G) \) be

\[
\nu(G) = \max_{(u, v) \in E} |N_u^\delta \cap N_v^\delta| \tag{38}
\]

Then, for any feasible transition \( x \to x^+ \),

\[
\nu(G) \geq \max_{i \in I} \Delta_i(x_i, x_i^+). \tag{39}
\]

Proof. Let \( x \to x^+ \) be a feasible transition. For any \( i \in I_{se}(x) \), \( \hat{U}_i^1 = \hat{U}_i^2 = 0 \). On the other hand, for any \( i \in I_{se}(x) \), the sampled partial utilities from the nodes \( N_u^\delta \cap N_v^\delta \), contribute equally to both \( \hat{U}_i^1 \) and \( \hat{U}_i^2 \). Hence,

\[
\max\{|N_u^\delta \cap N_v^\delta|, |N_u^\delta \cap N_v^\delta|\} \geq |\hat{U}_i^1 - \hat{U}_i^2|, \forall i \in I. \tag{40}
\]

In light of (40) and (28),

\[
\max\{|N_u^\delta \cap N_v^\delta|, |N_u^\delta \cap N_v^\delta|\} \geq \Delta_i(x_i, x_i^+), \forall i \in I. \tag{41}
\]

Since \( (a_i^1, a_i^2) \in E \) for any \( i \in I_{se}(x) \), (38) implies

\[
\nu(G) \geq \max\{|N_u^\delta \cap N_v^\delta|, |N_u^\delta \cap N_v^\delta|\}, \forall i \in I. \tag{42}
\]

Finally, (41) and (42) together imply (39).

Next, we show that \( r > \nu(G) \) is a sufficient condition to ensure that the paths between the states in \( X^0_R \) on a minimum resistance tree consist of unilateral experiments. As such, if \( r \) is sufficiently large, then simultaneous experiments become sufficiently rare as \( \epsilon \) goes down to zero.

Definition (Unilateral Experimental Path): A feasible sequence of states, \( P = \{x^1, x^2, \ldots, x^n\} \), is a unilateral experimental path if \( x^1, x^n \in X^0_R, x^2, \ldots, x^{n-1} \in X^0_T \) and for all \( 1 \leq p \leq n - 1 \)

\[
|I_{se}(x^p, x^{p+1})| = \begin{cases} 1 & \text{if } p = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{43}
\]

Lemma 5.3. Let \( T^* \) be a minimum resistance tree, and let \( x \to x^+ \in T^* \). If \( x \in X^0_R \), then \( |I_{se}(x, x^+)| = 1 \).

Proof. Since \( x \in X^0_R \), \( |I_{se}(x, x^+)| > 0 \), as otherwise, \( x^+ = x \) and \( x \to x^+ \) cannot be contained in a tree. Assume that \( |I_{se}(x, x^+)| > 1 \). Then, choose an arbitrary \( i \in I_{se}(x, x^+) \)}
to define an \( \tilde{x}^+ \neq x \) as
\[
\tilde{x}^+_j = \begin{cases} \frac{x^+_j}{x_i} & \text{if } j \neq i, \\ x_i & \text{otherwise.} \end{cases}
\] (44)

Note that \( x \to \tilde{x}^+ \) is a feasible transition, and \( R(x, \tilde{x}^+) = R(x, x^+) - r(|I_{se}(x, x^+)| - 1) \). Replacing \( x \to \tilde{x}^+ \) with \( x \to \tilde{x}^+ \) would give an alternative tree with a smaller resistance, which contradicts with \( T \) being a minimum resistance tree.

**Lemma 5.4.** Let \( x \to x^+ \in T^* \) for some minimum resistance tree \( T^* \). If \( x \in \mathcal{X}^0_R \) and \( r > \nu(G) \), then \( |I_{se}(x, x^+)| < |I_e(x)| \).

**Proof.** Since \( x \in \mathcal{X}^0_R \), \( I_{se}(x, x^+) = \emptyset \) doesn’t imply \( \tilde{x}^+ = x \).
Hence, there exists an \( \tilde{x}^+ \neq x \) such that \( x \to \tilde{x}^+ \) is feasible and \( I_{se}(x, \tilde{x}^+) = \emptyset \). For any such \( \tilde{x}^+ \), we have
\[
R(x, \tilde{x}^+) - R(x, x^+) \leq -r|I_{se}(x, x^+)| + |I_{es}(x, \tilde{x}^+)| \nu(G).
\] (45)

Note that \( |I_{es}(x, \tilde{x}^+)| \leq |I_e(x)| \). Hence, given \( r > \nu(G) \), the right side of (45) is negative for any \( |I_{se}(x, x^+)| \geq |I_e(x)| \).
In that case, replacing \( x \to x^+ \) with \( x \to \tilde{x}^+ \) would give an alternative tree with a smaller resistance, which contradicts with \( T \) being a minimum resistance tree. Consequently, \( |I_{se}(x, x^+)| < |I_e(x)| \).

**Lemma 5.5.** Let \( r > \nu(G) \), and let \( \mathcal{P} = \{x^1, x^2, \ldots, x^n\} \) be a sequence of states, where \( x^1, x^n \in \mathcal{X}^0_R \) and \( x^2, \ldots, x^{n-1} \in \mathcal{X}^T_R \). If \( \mathcal{P} \in T^* \) for some minimum resistance tree \( T^* \), then \( \mathcal{P} \) is a unilateral experimentation path.

**Proof.** Since \( x^1 \in \mathcal{X}^0_R \), from Lemma 5.3, we have \( |I_{se}(x^1, x^2)| = 1 \) leading to \( |I_e(x^2)| = 1 \). Furthermore, for \( r > \nu(G) \), from Lemma 5.4, we have \( |I_{se}(x^2, x^3)| = 0 \). Hence, we have \( |I_e(x^3)| \leq 1 \). Using Lemma 5.4 recursively along \( \mathcal{P} \) we obtain
\[
|I_{se}(x^p, x^{p+1})| = \begin{cases} 1 & \text{if } p = 1, \\ 0 & \text{otherwise.} \end{cases}
\] (46)

Hence, \( \mathcal{P} \) is a unilateral experimentation path.

**Lemma 5.6.** If \( \mathcal{P} = \{x^1, x^2, \ldots, x^n\} \) be a unilateral experimentation path, then
\[
R(\mathcal{P}) = \sum_{p=1}^{n-1} R(x^p, x^{p+1}) = r + \max\{\phi(x^n), \phi(x^1)\} - \phi(x^n).\]
(47)

**Proof.** Since \( \mathcal{P} = \{x^1, x^2, \ldots, x^n\} \) be a unilateral experimentation path, for \( x^p, x^{p+1} \in \mathcal{X}^T_R \), we have
\[
|I_{se}(x^p, x^{p+1})| = |I_{es}(x^p, x^{p+1})| = 0.
\] (48)

Hence, such transitions have zero resistance, resulting in
\[
R(\mathcal{P}) = R(x^1, x^2) + R(x^{n-1}, x^n).
\] (49)

Note that, since \( x^1 \in \mathcal{X}^0_R \) and \( \mathcal{P} \) is a unilateral experimentation path, we have \( R(x^1, x^2) = r \) and \( R(x^{n-1}, x^n) = \Delta_i(x^{n-1}, x^n) \), where \( i \in I \) is the unique experimenting agent. Since all the other agents are stationary, i.e. \( a_{-i} \) is constant along \( \mathcal{P} \), the estimated utilities satisfy
\[
(\hat{U}_1)^{n-1} = \sum_{v \in \mathcal{X}^0_R} u(v, a_{-i}) = U_i(a_i^1, a_{-i}),
\] (50)
\[
(\hat{U}_2)^{n-1} = \sum_{v \in \mathcal{X}^0_R} u(v, a_{-i}) = U_i(a_i^2, a_{-i}).
\] (51)

Plugging (50) and (51) into (28) we obtain
\[
\Delta_i(x_i^{n-1}, x_i^n) = \max\{U_i(x^n), U_i(x^1)\} - U_i(x^n).
\] (52)

Since \( \Gamma_{DGC} \) is a potential game, from (52) we obtain
\[
\Delta_i(x_i^{n-1}, x_i^n) = \max\{\phi(x^n), \phi(x^1)\} - \phi(x^n).
\] (53)

**Lemma 5.7.** Let \( r > \nu(G) \), and let \( T_{x^*}^\prime \) and \( T_{x^*}^\prime \) be minimum resistance trees rooted at some \( x, x^* \in \mathcal{X}^0_R \). Then,
\[
R(T_{x^*}^\prime) \leq R(T_{x^*}^\prime) \Rightarrow \phi(x) \geq \phi(x^*).
\] (54)

**Proof.** For \( r > \nu(G) \), in light of Lemma 5.5, the paths between the states in \( \mathcal{X}^0_R \) on a minimum resistance tree consist of unilateral experimentations. Let \( x_0^0 \in \mathcal{X}^0_R \) and let \( T_{x^0_R} \) be a minimum resistance tree rooted at \( x_0^0 \). Let \( x^p_R \in \mathcal{X}^0_R \) be a state such that \( R(T_{x^0_R}) \leq R(T_{x^p_R}) \) and the unique path, \( \mathcal{P} \in T_{x^p_R} \), from \( x^0_R \) to \( x^p_R \) consists of \( n \) unilateral experimentations, i.e.
\[
R(\mathcal{P}) = \sum_{k=1}^{n} R(P_k),
\] (55)

where \( P_k \) is the unilateral experimentation starting at \( x^0_R \) and ending at \( x^p_R \). Note that, for each such \( P_k \), there exists a feasible unilateral experimentation path \( P_k^* \) in the reversed direction, starting at \( x^p_R \) and ending at \( x^0_R \). Replacing each \( P_k \) with \( P_k^* \), one can construct a tree rooted at \( T_{x^p_R} \). Note that the resistances of these trees satisfy
\[
R(T_{x^p_R}) - R(T_{x^p_R}) = \sum_{k=1}^{n} (R(P_k) - R(P_k^*)) = \sum_{k=1}^{n} (\phi(x^p_R) - \phi(x^{n-k+1})) = \phi(x^0_R) - \phi(x^p_R).
\] (56)

Note that by definition \( R(T_{x^p_R}) \leq R(T_{x^p_R}) \). Hence, if \( R(T_{x^p_R}) \leq R(T_{x^p_R}) \), then \( R(T_{x^p_R}) \leq R(T_{x^p_R}) \) for any \( T_{x^p_R} \).

Plugging this into (56), we obtain \( \phi(x^0_R) \leq \phi(x^p_R) \).

**Theorem 5.8.** Let \( G = (V, E) \) be a connected graph, let all agents follow the CFCM algorithm with \( r > \nu(G) \), and let \( x \) be a stochastically stable state of the resulting Markov chain. Then, \( I_n(x) = I \) and \( |V_c(x)| = |V_c(x^r)| \) for all \( x^r \in X \).

**Proof.** Let \( x \) be a stochastically stable state of the resulting Markov chain. Due to Lemma 3.2, \( x \in \mathcal{X}^0_R \) and \( R(T_x^\prime) \leq R(T_x^\prime) \) for all \( x^r \in \mathcal{X}^0_R \). Hence, due to (36), \( I_n(x) = I \).
In light of Lemma 5.7, if \( r > \nu(G) \), then \( R(T_x^\prime) \leq R(T_x^\prime) \) implies \( \phi(x) \geq \phi(x^r) \) for all \( x^r \in \mathcal{X}^0_R \). Since \( \phi(x) = |V_c(x)| \) for \( \Gamma_{DGC} \) and \( V_c(x) \) only depends on the current positions of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2016.2518083, IEEE Transactions on Control of Network Systems
The agents on the graph, $\phi(x) \geq \phi(x')$ for all $x' \in X^0_r$ implies $|V_c(x)| \geq |V_c(x')|$ for all $x' \in X$.

Theorem 5.8 indicates that if all agents follow the CFCM algorithm with sufficiently large $r$, then the stochastically stable states are all-stationary states maximizing the number of covered nodes. As such, the agents asymptotically maintain maximum coverage with an arbitrarily high probability for arbitrarily small values of the noise parameter $\epsilon$.

C. Adaptive Implementation

As presented in the previous section, a sufficient condition for $r$ to guarantee the stochastic stability of the coverage maximizers is that $r > \nu(G)$. Accordingly, an appropriate $r$ can be determined if $\nu(G)$ or any graph property that defines an upper bound of $\nu(G)$ is available as prior information. When no prior information about the target graph is available, one possible way of achieving the same limiting behavior is to let each agent have an individual $r_i$ that is updated based on the local observations as the agent moves on the graph. In this section, we present such an adaptive implementation of the CFCM algorithm that achieves the desired limiting behavior by having every agent $i$ eventually obtain a stationary $r_i = \beta + \nu(G)$ for some $\beta > 0$.

Algorithm III: Adaptive CFCM

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>initialization: $\epsilon \in \mathbb{R}^+$ (small), $\beta \in \mathbb{R}^+$, $a_i \in A$, arbitrary. $S_i = {a_i}$, $k_i = 1$, $U^1_i = U^2_i = 0$, $r_i = \beta$.</td>
</tr>
<tr>
<td>2</td>
<td>repeat</td>
</tr>
<tr>
<td>3</td>
<td>$a_i = S_i(k_i)$, $a^1_i = S_i(1)$, $a^2_i = S_i(\lfloor</td>
</tr>
<tr>
<td>4</td>
<td>if ($</td>
</tr>
<tr>
<td>5</td>
<td>Generate a random (uniform) $\gamma \in [0, 1]$.</td>
</tr>
<tr>
<td>6</td>
<td>if ($\gamma \leq \epsilon^r$)</td>
</tr>
<tr>
<td>7</td>
<td>$a^2_i$ is randomly (uniform) chosen over $A^r_i(a^1_i)$.</td>
</tr>
<tr>
<td>8</td>
<td>$r_i = \max(r_i, \beta +</td>
</tr>
<tr>
<td>9</td>
<td>$S_i = E(a^1_i, a^2_i)$.</td>
</tr>
<tr>
<td>10</td>
<td>end if</td>
</tr>
<tr>
<td>11</td>
<td>else</td>
</tr>
<tr>
<td>12</td>
<td>if ($k_i \geq k$; $\forall k \in {k \mid S_i(k) = a_i}$)</td>
</tr>
<tr>
<td>13</td>
<td>$U^1_i = U^1_i + u_i(a_i, a_i)$; if $a_i \in A^r_i$.</td>
</tr>
<tr>
<td>14</td>
<td>$U^2_i = U^2_i + u_i(a_i, a_i)$; if $a_i \in A^2_i$.</td>
</tr>
<tr>
<td>15</td>
<td>end if</td>
</tr>
<tr>
<td>16</td>
<td>if ($k_i =</td>
</tr>
<tr>
<td>17</td>
<td>$a = \epsilon^{-U^1_i}$, $\beta = \epsilon^{-U^2_i}$.</td>
</tr>
<tr>
<td>18</td>
<td>$S_i = \left{ {a_j^1} \text{ w.p. } \frac{U^1_i}{U^1_i + U^2_i}, \right} {a_j^2}$ otherwise.</td>
</tr>
<tr>
<td>19</td>
<td>$k_i = 1$, $U^1_i = U^2_i = 0$.</td>
</tr>
<tr>
<td>20</td>
<td>else</td>
</tr>
<tr>
<td>21</td>
<td>$k_i = k_i + 1$.</td>
</tr>
<tr>
<td>22</td>
<td>end if</td>
</tr>
<tr>
<td>23</td>
<td>end if</td>
</tr>
<tr>
<td>24</td>
<td>end repeat</td>
</tr>
</tbody>
</table>

The adaptive implementation differs from the CFCM algorithm by having each $r_i$ as a state variable rather than a constant. In analyzing the resulting Markov chain, we will use the same notation as before and the distinction should be clear from the context. Accordingly, the state of each agent $i$ is denoted as $x_i = [S_i, k_i, U^1_i, U^2_i, r_i]$, whereas the global state is the concatenation of the individual states, $x = [x_1, x_2, \ldots, x_m]$, and $X$ denotes the state space. Since each $r_i$ is monotonically increasing (see line 8 in Algorithm III), the Markov chain induced by the adaptive implementation is not irreducible unless $\nu(G) = 0$. However, we will show that it has a unique limiting distribution that behaves the same as the limiting distribution under CFCM as $\epsilon$ goes down to zero.

Theorem 5.9. Let $G = (V, E)$ be a connected graph, and let all agents follow the adaptive CFCM algorithm. Then, the resulting Markov chain has a unique limiting distribution $\mu_c^*$ such that

$$\lim_{\epsilon \to 0^+} \mu^*_c(x) > 0 \Rightarrow x \in X^*$,

where

$$X^* = \{x \in X \mid r_i = \beta + \nu(G), \forall i \in I, I_a(x) = I, |V_c(x)| \geq |V_c(x')|, \forall x' \in X\}.$$  

Proof. Let $G = (V, E)$ be a connected graph, and let $X_c$ be the set of states such that each agent $i$ has $k_i = \beta + \nu(G)$, i.e.,

$$X_c = \{x \in X \mid r_i = \beta + \nu(G), \forall i \in I\}.$$  

First, we will show that $X_c$ is the unique closed communicating class of the Markov chain, i.e., any $x \notin X_c$ is transient, any state in $X_c$ is reachable from any other state in $X_c$, and the probability of leaving $X_c$ is zero. Let $(v, v') \in E$ be such that $|X^0_v \setminus X^0_{v'}| = \nu(G)$. Note that, for any $\epsilon, r_i > 0$, any stationary agent $i$ has a non-zero probability to experiment and pick any of the neighboring nodes as its next stationary position. Hence, for a connected graph $G$, any agent $i$ such that $r_i \neq \beta + \nu(G)$ can eventually become stationary at $a^2_i = v$ and start an experiment by picking $a^2_i = v'$. Note that each $r_i$ is monotonically increasing under the adaptive CFCM (line 8), and such a feasible trajectory would result in $r_i = \beta + \nu(G)$. Furthermore, $\beta + \nu(G)$ is the maximum value that any $r_i$ can attain under the adaptive CFCM due to the definition of $\nu(G)$ in (38). Hence, $X_c$ is reachable from any $x \notin X_c$, and once inside $X_c$, the probability of leaving is zero. Accordingly, any $x \notin X_c$ is a transient state. Note that the behavior of the Markov chain inside $X_c$ is identical to the chain induced by CFCM with the same value of $\epsilon$ and $r = \beta + \nu(G)$. Hence, any state in $X_c$ is reachable from any other state in $X_c$. Consequently, $X_c$ is the unique closed communicating class.

Any finite Markov chain with a unique closed communicating class is known to have unique stationary distribution that is determined by the transition probabilities within the closed communicating class. Hence, the chain induced by the adaptive CFCM has a unique limiting distribution $\mu^*_c$ such that $\mu^*_c(x) = 0$ for any $x \notin X_c$. Furthermore, for any $x \in X_c$, it follows from Theorem 5.8 that if $\lim_{\epsilon \to 0^+} \mu^*_c(x) > 0$, then
\[ I_s(x) = I \text{ and } |V_c(x)| \geq |V_c(x')| \text{ for all } x' \in X_c. \] Since \( V_c(x) \) is determined by only the agent positions, \( |V_c(x)| \geq |V_c(x')| \) for all \( x' \in X \). Consequently, if \( \lim_{t \to 0^+} \mu_c^t(x) > 0 \), then \( x \in X^* \).

In light of Theorem 5.9, if all the agents follow the adaptive CFCM algorithm on any connected graph \( G \), then they asymptotically maintain maximum coverage with an arbitrarily high probability for arbitrarily small values of the noise parameter \( \epsilon \). The necessity of choosing a sufficiently large \( r \) is removed for the adaptive implementation since the agents learn a suitable value as they move on the graph.

**VI. SIMULATION RESULTS**

In this section, some simulation results are presented to demonstrate the performance of the proposed method. In the simulations, a group of 13 agents are initially placed at an arbitrary node of a connected random geometric graph. Each agent has a sensing range \( \delta = 1 \). The graph consists of 45 nodes and 62 edges, and it has \( \nu(G) = 2 \).

In the first simulation, the agents follow the adaptive CFCM algorithm with \( \epsilon = 0.03 \) and \( \beta = 0.01 \). The number of covered nodes throughout a period of 150000 time steps is shown in Fig. 5, whereas the configuration of the agents on the graph at some instants is provided in Fig. 6. As depicted in Fig. 5, after a sufficient amount of time, the agents maintain complete coverage with a very high probability. For \( 90000 \leq t \leq 150000 \), the average number of covered nodes at each time step is computed as 44.77.

![Fig. 5. The number of covered nodes as a function of time (adaptive CFCM).](image)

In the second simulation, we consider the same scenario for the agents following the CFCM algorithm. In CFCM, the parameter \( r \) sets the likelihood of each stationary agent to update its action. Accordingly, increasing \( r \) also increases the mixing time of the resulting Markov chain. Hence, in practice, it is desirable to choose \( r \) as small as possible while ensuring the stochastic stability of the optimal configurations. Note that a suitable choice of \( r \) depends on the graph structure, and we provided \( r > \nu(G) \) as a sufficient condition. However, it may not be necessary in many cases since simultaneously updating agents do not necessarily influence the utility estimations of each other, especially when they are sufficiently far from each other. In this simulation, we demonstrate such a case. The agents start at the same initial condition as the previous simulation, and they follow the CFCM algorithm with \( \epsilon = 0.03 \) and \( r = 1.6 \). The number of covered nodes throughout a period of 75000 time steps is shown in Fig. 7, whereas the configuration of the agents on the graph at some instants are provided in Fig. 8. For \( 45000 \leq t \leq 75000 \), the average number of covered nodes at each time step is computed as 44.48.

![Fig. 7. The number of covered nodes as a function of time (CFCM).](image)

Finally, in order to compare the performance with a setting
that allows for communications, we also present a simulation of the same scenario with BLLL. The agents start at the same initial condition as the previous simulations, and BLLL is executed with the same noise parameter $\epsilon = 0.03$. The number of covered nodes throughout a period of 5000 time steps is shown in Fig. 9, whereas the configuration of the agents on the graph at some instants are provided in Fig. 10. As illustrated in Fig. 9, after a sufficient amount of time, the agents maintain complete coverage with a very high probability. For $3000 \leq t \leq 5000$, the average number of covered nodes at each time step is computed as 44.64.

**Fig. 9.** The number of covered nodes as a function of time (BLLL).

**Fig. 10.** The configuration of 13 agents on the graph at some instants of the simulation (BLLL). The nodes occupied by at least one agent are black, the nodes covered by at least one agent are gray, and the nodes that are not covered are white.

The simulation results show that the agents achieve optimal coverage in all three cases. When the agents are allowed to communicate, they can maximize the coverage much faster as expected. Despite the slower convergence to the limiting distribution, the main advantage of the CFCM algorithm is that the agents do not require any explicit communications, and optimal coverage is achieved if the parameter $r$ is selected sufficiently large. Furthermore, the adaptive implementation of CFCM achieves optimal coverage without relying on the apriori selection of a suitable $r$. Accordingly, CFCM or its adaptive implementation can be employed to optimally distribute some mobile resources on networks, even in scenarios that do not allow for explicit communications.

**VII. Conclusions**

In this paper, a game theoretic approach was proposed for distributed coverage of networked systems by mobile agents with local capabilities. We considered a distributed graph coverage (DGC) problem, where the network is modeled as an undirected graph, and the agents are located on some nodes of the graph. Each agent can sense the graph structure and the presence of the other agents within its $\delta$-neighborhood, where $\delta$ is the sensing range. Any node of the graph is covered if it is within the sensing range of at least one agent. The agents move locally on the graph, and they aim to maximize the number of covered nodes. We studied this problem particularly for agents with no explicit communications among themselves.

A game theoretic formulation of the DGC problem was obtained by designing a potential game, $\Gamma_{DGC}$. In $\Gamma_{DGC}$, the action of each agent is defined as its position on the graph, and the utility of each agent is equal to the number of nodes covered only by itself. It was shown that $\Gamma_{DGC}$ can be paired with a learning algorithm such as BLLL to maximize the coverage. However, such learning algorithms require the agents to measure their current utilities. In $\Gamma_{DGC}$, the actual utilities can not be computed without explicit communications since the agents with overlapping coverage are not necessarily within the sensing range of each other. In order to address this issue, we presented a communication-free learning algorithm, namely the CFCM. In CFCM, the agents follow a noisy best-response policy based on the estimated utilities gathered by moving around their current positions. The algorithm has a noise parameter, $\epsilon \in \mathbb{R}^+$, and a second parameter, $r \in \mathbb{R}^+$, that sets the likelihood of remaining stationary. We showed that the CFCM algorithm induces a regular perturbed Markov chain and the stochastically stable states are the coverage maximizers for sufficiently large values of $r$. A sufficient value of $r$ was derived from the topology of the graph. Furthermore, we presented an adaptive implementation of CFCM such that the agents learn a suitable value of $r$ based on their local observations as they move on the graph. As such, they achieve maximum coverage on any connected graph, without any prior information. Some simulation results were also presented to demonstrate the performance of the proposed method.

**References**


