A Flocking-Based Approach to Maintain Connectivity in Mobile Wireless Ad Hoc Networks

Abdullah Konak, George Buchert, James Juro
Penn State Berks, Tulpehocken Road P.O. Box 7009 Reading, PA 19610-6009 USA

Abstract

Mobile Ad Hoc Networks (MANET) have a set of unique challenges, particularly due to mobility of nodes, that need to be addressed to realize their full potentials. Because the mobile nodes of a MANET are free to move rapidly and arbitrarily, the network topology may change unexpectedly. This paper presents a decentralized approach to maintain the connectivity of a MANET using autonomous, intelligent agents. Autonomous agents are special mobile nodes in a MANET, but unlike other nodes, their function is to proactively prevent network bottlenecks and service problems by intelligently augmenting the network topology. To achieve this function without depending on a central network management system, autonomous agents are expected to dynamically relocate themselves as the topology of the network changes during the mission time. A flocking-based heuristic algorithm is proposed to determine agent locations. A computational study is performed to investigate the effect of basic flocking behaviors on the connectivity of a MANET.

Keywords: Ad hoc networks, swarm intelligence, flocking algorithms, network connectivity

1. Introduction

There are two types of wireless networks: infrastructure dependent and infrastructure-less networks. Infrastructure dependent networks, such as cellular networks, rely on base stations to route packets. Such networks are...
referred to as ‘single hop’ networks, as information is sent from the trans-
mittting node to a base node (e.g., tower), then onto the receiving node. Routing is simplified in an infrastructure dependent network since a single 
central location keeps track of node locations and handles routing protocol. Infrastructure-less ad hoc networks have no central base station to relay data packets. Instead, each node is capable of routing and retransmitting data packets. Such a network is sometimes referred to as a ‘multi-hop’ network [16] as packets may be routed through several nodes before reaching their des-
tinations. Nodes in a multi-hop network must be more intelligent than their infrastructure dependent counterparts as routing and resource management functions are distributed over the network.

Mobile wireless ad hoc networks (MANET) are instantaneous, autonomous multi-hop networks that provide service to users wherever and whenever the service is needed. Development time and cost of MANET are also relatively low compared to infrastructure dependent networks. Because of the relative ease with which MANET can be established, the technology has found use in situations where establishing an appropriate network infrastructure would be cost or time prohibitive, such as in military, emergency, or search and rescue operations. MANET also have the potential to create ubiquitous communication networks by seamlessly interconnecting thousands of devices [1, 5, 21].

The flexible nature of MANET, particularly mobility of nodes, brings a set of unique challenges that need to be addressed to realize their full potential. Because mobile nodes in MANET are free to move rapidly and arbitrarily, the network topology may change unexpectedly. The channel capacities of wireless links are limited and depend on the distance between nodes and environmental factors. Therefore, unexpected network bottlenecks may occur as the network topology and link capacities dynamically change over the time. In addition, certain devices may have limited power and processing capability. All these challenges may impair the performance of MANET.

Several approaches have been proposed in the literature to address the problems in MANET due to unpredictable node movements. One of the ma-
ajor problems is the accessibility of the centralized network services used by all nodes when the network is disconnected. This problem can be addressed by replicating network services [25] or critical data [11] at multiple nodes and dynamically deploying these nodes to disconnected partitions of the network. Another problem is the delivery of data packets across disconnected network
partitions. In the literature, special agent nodes are proposed to address this problem [6, 27]. These agent nodes can buffer packets until their destination nodes are reachable. When agent nodes are connected to a network partition, they deliver their payload. Alternatively, several papers [8, 15] propose network topology control by modifying node trajectories or power levels [13, 24].

There has been very limited work in the literature to improve network connectivity in MANET through mobile agents. For example, Ou et al. [20] propose special relay nodes that can adjust their locations to assist disconnected network partitions. Chadrashekar et al. [4] define the problem of achieving connectivity in disconnected ground MANET by dynamically placing unmanned air vehicles (UAVs) which function as relay nodes. Zhu et al. [28] also propose using UAVs equipped with communication capabilities to provide services to ground-based MANET. Recently, Hauert et al. [9] propose the deployment of a swarm of UAVs for search and rescue missions. During a mission time, UAVs are expected to maintain direct or indirect connection to their base-station through the ad hoc network that they form. Recently, Konak et al. [12] and Dengiz et al. [7] propose a MANET management system to improve MANET connectivity using agent nodes managed by a centralized network management system. In this MANET management system, a particle swarm optimization algorithm is proposed to dynamically determine agent deployment decisions by assuming that the global state of a network is available, and agent deployment decisions can be communicated to agents at all times.

As briefly summarized above, the current research on improving network performance and connectivity in MANET through auxiliary nodes, which are called agents in this paper, assumes the existence of a central network management system. This central management system is assumed to be aware of the global state of the network and capable of communicating with agents at all times. However, this assumption is not realistic in many real-world MANET. The research in this paper takes a different direction and proposes autonomous agent nodes, each of which is capable of making deployment decisions independently without relying on a central network management system. The proposed approach is to devise a set of rules that are similar to the flocking rules [18, 22] to create a group-flocking behavior from the individual behaviors of a set of autonomous agents. Recently, there has been increased interest in multi-agent flocking for automated control in sensor networks and UAVs (e.g., [2, 10, 14, 19, 17, 23, 26]). However, multi-agent
flocking has not been previously applied to improve network connectivity in MANET.

2. Problem Formulation and Assumptions

Let $G(t) = (N(t), E(t))$ denote a MANET with node set $N(t)$ and edge set $E(t)$ at time $t$. Node set $N(t)$ includes two types of nodes: users ($U(t)$) and agents ($A(t)$). Users demand network services, and they are assumed to move freely. Agents are responsible for helping users experience the best network service possible by dynamically adjusting their locations. Edge set $E(t)$ depends on the locations of users and agents, their transmission ranges, as well as other environmental factors at time $t$. Given position $p_i(t) = (x_i(t), y_i(t))$ and transmission range $R_i$ of each node $i \in N(t)$, edge set $E(t)$ at time $t$ is determined as follows:

$$E(t) = \{(i, j) : i, j \in N(t), i \neq j, d_{ij} \leq \min(R_i, R_j)\} \quad (1)$$

where $d_{ij} = \|p_i(t) - p_j(t)\|$ (it is assumed that $(i, j) \equiv (j, i)$, and $\|\cdot\|$ denotes the Euclidean norm of a vector).

The overall objective is to maximize the connectivity of users during a mission time by dynamically locating a set of agents in response to the changes in the network topology. Given network topology $G(t)$ at time $t$, let $Q(G(t))$ be the percent of the connected (directly or indirectly) user node pairs. If the mission time is sliced into $T$ discrete time steps, and given that agent $i$ can travel a maximum of $V_i$ unit distance in a time step, the overall problem can be expressed as follows:

Problem Agent:

$$\text{Max } Q = \frac{1}{T} \sum_{t=1}^{T} Q(G(t))$$

$$\|p_i(t) - p_i(t - 1)\| \leq V_i \quad \forall i \in A(t), t = 1, \ldots, T$$

Although Problem Agent is useful for describing the overall optimization problem, it is not practical to be used in real-world MANET for several reasons. First of all, Problem Agent assumes an implicit centralized network management system that is aware of the global state of the network (i.e., $G(t)$ is known by all agents) and that can communicate with all agents at all times. These assumptions may hold for some cases, but they are not realistic for many real-world MANET. More importantly, the future locations of users are
not known when the deployment decision is made at any time $t$. Therefore, Problem Agent represents the theoretical best case of what can be achieved in the discrete time domain. In addition, Problem Agent is a non-linear mixed-integer programming problem, which is very difficult to optimally solve with limited CPU resources available to agent nodes. Furthermore, deployment decision should be made in very short time intervals.

The overall objective of the proposed flocking-based heuristic is to dynamically determine the best possible locations of autonomous agents and deploy them accordingly to maximize MANET connectivity: (i) in the absence of a centralized network management system; (ii) without complete and up-to-date information about the global state of the network; and (iii) with inadequate computing resources or limited time to carry out computationally expensive calculations. This paper proposes a decentralized approach based on the concepts from swarm intelligence, particularly based on the earlier work on flocking agents [22]. In the proposed agent deployment approach, agents use simple rules to determine their new locations as the topology of the network randomly changes. Preliminary results from a simulation study to determine the effects of agent level behaviors on the global network connectivity are reported. In addition, the performance of the proposed flocking-based heuristic is compared to a quadratically constrained mixed-integer programming approach.

3. Proposed Approach

Flocking is a collective behavior of independent but interacting agents. Early work on flocking and swarm theory focused on mimicking realistic movements of flocking animals. Reynolds [22] introduced three basic rules that achieved the first simulated flocking in computer animations as follows:

- cohesion: attempt to stay close to nearby flockmates.
- separation: avoid collisions with nearby flockmates.
- alignment: attempt to match velocity with nearby flockmates.

In Reynolds’s model, each agent can make independent decisions to maintain the flock. In addition, it is assumed that individual agents have no knowledge of the wider arrangement of other agents, and instead they are only aware of those agents in their immediate vicinity. The problem studied
in this paper is quite different than the basic flocking behavior. Therefore, these basic flocking rules are redefined to maintain the connectivity of users.

Let vector $v_i(t) = (vx_i(t), vy_i(t))$ denote the velocity of agent $i$ at time $t$. Velocity vector $v_i(t)$ represents the direction and the distance that agent $i$ intends to move between times $t$ and $t + 1$, and the new position of the agent at the beginning of time $t + 1$ is given as follows:

$$p_i(t + 1) = p_i(t) + \min(V_i, \|v_i(t)\|)\frac{v_i(t)}{\|v_i(t)\|}$$  \hfill (2)

where $V_i$ represents the maximum distance that agent $i$ can move in a time period. Equation (2) states that agent $i$ can move a maximum of $V_i$ unit distance at the direction of $v_i(t)$. Velocity vector $v_i(t)$ is updated based on three agent behaviors, separation, cohesion, and exploration as described in the following sections.

Although the proposed approach in this paper stems from the earlier research on flocking or target tracking, the formulated problem is quite different. Firstly, users are free to move at their will. Secondly, a MANET is expected to have many user nodes (targets) compared to limited number of agents (trackers). Therefore, agent behaviors are modified to consider these facts. In the proposed approach, the separation and cohesion behaviors of an agent depend on the spatial neighbors of the agent. The sets of agent $i$’s user ($NU_i(t)$) and agent ($NA_i(t)$) spatial neighbors at time $t$ are defined as follows:

$$NU_i(t) = \{j \in U(t) : d_{ij} < \min(R_i, R_j), i \neq j\}$$  \hfill (3)

$$NA_i(t) = \{j \in A(t) : d_{ij} < \min(R_i, R_j), i \neq j\}$$  \hfill (4)

3.1. Separation Behavior

Separation behavior encourages an agent to move away from other nodes in its neighborhood if the agent is too close to them. The separation velocity of agent $i$ with respect to agent $j$ is given as follows:

$$v_{sa_{ij}}(t) = \begin{cases} 
0.5(S_A - d_{ij})(p_i(t) - p_j(t))/d_{ij} & d_{ij} \leq S_A, \\
0 & \text{otherwise},
\end{cases}$$

where $S_A$ is the target distance that should be maintained between two agents. Essentially, if the distance between two agents $i$ and $j$ is less than
agent \(i\) moves away from agent \(j\) by the half length of \((S_A - d_{ij})\) in the opposite direction. Note that because agent \(j\) also exhibits the same behavior with respect to agent \(i\), only the half distance is considered in equation (5). The separation velocity of agent \(i\) with respect to user \(j\) is given as follows:

\[
\text{vsu}_{ij}(t) = \begin{cases} 
(S_U - d_{ij})(p_i(t) - p_j(t))/d_{ij} & d_{ij} \leq S_U, \\
0 & \text{otherwise.} 
\end{cases} 
\]  

(6)

where \(S_U\) is the target distance between an agent and a user. Finally, the total velocity of agent \(i\) due to separation behavior is given as follows:

\[
\text{vs}_i(t) = \sum_{j \in N_{A_i}(t)} \text{vs}_{aij}(t) + \sum_{j \in N_{U_i}(t)} \text{vs}_{uij}(t) 
\]  

(7)

### 3.2. Cohesion Behavior

Cohesion is the opposite of separation. Cohesion behavior encourages an agent to move closer to other nodes. The cohesion velocity of agent \(i\) with respect to agent \(j\) is given as follows:

\[
\text{vca}_{ij}(t) = \begin{cases} 
0.5(d_{ij} - S_A)(p_j(t) - p_i(t))/d_{ij} & d_{ij} \geq S_A, \\
0 & \text{otherwise.} 
\end{cases} 
\]  

(8)

and the cohesion velocity of agent \(i\) with respect to user \(j\) is defined as follows:

\[
\text{vc}_{ij}(t) = \begin{cases} 
(d_{ij} - S_U)(p_j(t) - p_i(t))/d_{ij} & d_{ij} \geq S_U, \\
0 & \text{otherwise.} 
\end{cases} 
\]  

(9)

Finally, the total velocity of agent \(i\) due to cohesion behavior is given as follows:

\[
\text{vc}_i(t) = \sum_{j \in N_{A_i}(t)} \text{vca}_{ij}(t) + \sum_{j \in N_{U_i}(t)} \text{vc}_{uij}(t) 
\]  

(10)

### 3.3. Exploration Behavior

Because users move randomly, an agent or a group of agents can be isolated from other nodes. In such cases, separation and cohesion behaviors are not applicable. Exploration encourages agents to move randomly with an
aim of discovering other nodes. The movement of agent $i$ due to exploration behavior is given as follows:

$$\mathbf{v}_{ei}(t) = (V_i \cos(\theta), V_i \sin(\theta))$$

(11)

where $\theta = \text{Rand}(0, 2\pi)$. Essentially, agent $i$ moves away from its current position as much as possible in the random direction indicated by angle $\theta$ (in radians).

3.4. Overall Agent Behaviors

Each agent $i$ updates its velocity at the beginning of time $t$ based on sets $\mathcal{NA}_i(t)$ and $\mathcal{NU}_i(t)$. Two cases are considered as follows:

**Case 1:** $\mathcal{NA}_i(t) = \emptyset$ and $\mathcal{NU}_i(t) = \emptyset$

In this case, agent $i$ is only in the exploration mode because it cannot exhibit the separation or cohesion behaviors. Therefore, the movement of agent $i$ solely depends on $\mathbf{v}_{ei}(t)$, and its current velocity is determined as follows:

$$\mathbf{v}_i(t) = \alpha \mathbf{v}_{ei}(t) + (1 - \alpha)\mathbf{v}_i(t - 1)$$

(12)

where smoothing parameter $\alpha$ (between 0 and 1) indicates how much previous information gathered by an agent is incorporated into its new deployment decision. Note that current velocity of an agent may have been formed by previous the separation or cohesion behaviors. Therefore, the current velocity of an agent may include important information about the previous interactions of the agent. Therefore, smoothing parameter $\alpha$ can be considered as a memory parameter.

**Case 2:** $\mathcal{NA}_i(t) \neq \emptyset$ or $\mathcal{NU}_i(t) \neq \emptyset$

In this case, agent $i$ updates its velocity as follows:

$$\mathbf{v}_i(t) = \alpha (w_s \mathbf{v}_{si}(t) + w_c \mathbf{v}_{ci}(t) + w_e \mathbf{v}_{ei}(t))$$

$$+(1 - \alpha)\mathbf{v}_i(t - 1)$$

(13)

where $\mathbf{v}_{si}(t)$, $\mathbf{v}_{ci}(t)$, and $\mathbf{v}_{ei}(t)$ are the velocities of agent $i$ due to separation, cohesion, and exploration behaviors, respectively, and $w_s$, $w_c$, and $w_e$ are their corresponding weights (between 0 and 1). The new velocity of an agent is a combination of its current velocity and the new information gained by
its separation, cohesion, and exploration behaviors. By setting weights $w_s$, $w_c$, and $w_e$, various agent behaviors can be modeled.

In summary, the proposed agent behavior attempts to maintain the distance of $S_A$ between two agents and the distance of $S_U$ between an agent and a user through the separation and cohesion behaviors, and to create new connections through the exploration behavior. The proposed approach is very straightforward, and it can be implemented in a decentralized manner. More complex agent behaviors can be incorporated into the proposed agent deployment strategy as well. However, initial experiments have shown that the three agent behaviors defined above are highly effective in maintaining the network connectivity. In Fig. 1 and Fig. 2, the topology of a MANET is illustrated at the beginning and the end of a simulation scenario, respectively, where all network nodes are agents. Although this scenario does not include any users, it is relevant for demonstrating how the separation, cohesion, and exploration behaviors can lead to a network topology that seems to be guided by a global network management system. As seen in Fig. 2, all agents are able to get connected and maintain the global network connectivity. In the following section, these behaviors are studied under various scenarios.
4. Computational Experiments

Computational experiments focus on answering two research questions: (i) What are the relationships between the agent behaviors and the global network connectivity? and (ii) What are the best agent behaviors to improve the network connectivity in various scenarios? A simulation study was carried out using 12 different network configurations, including all combinations of 10, 20, and 30 users and 2, 3, 4, and 5 agents. The details of the simulation are described in the following section. The numbers of users and agents are constant during a simulation run (i.e., \( nu = |U(t)| \) and \( na = |A(t)| \)). In Table 1, Fig. 4, and Table 2 test networks are identified by pair \((nu,na)\) where \(nu\) and \(na\) are the number of users and agents, respectively.

4.1. MANET Simulation Environment

The simulation area is a circle with the center coordinates of (300, 300) and radius of 300 unit distance. The initial locations and initial velocities of all nodes are randomly determined. The movements of user nodes are simulated using a version of Random Waypoint Mobility Model [3] over a mission time of \( T \) periods. Within a time period, each user \(i\) travels a random
distance between 0 and $V_i$ in the direction of angle $\theta_i(t)$, and each user $i$ randomly changes its direction with a probability of $\rho$ as follows:

$$\theta_i(t) = \begin{cases} 
\text{Rand}(0, 2\pi) & \text{Rand}(0, 1) \leq \rho, \\
\theta_i(t - 1) & \text{otherwise.}
\end{cases}$$

The new position of user $i$ is given by

$$\mathbf{p}_i(t + 1) = \mathbf{p}_i(t) + \text{Rand}(0, V_i)(\cos(\theta_i(t)), \sin(\theta_i(t)))$$

If new position $\mathbf{p}_i(t + 1)$ of user $i$ is outside of the simulation area, $\theta_i(t)$ is randomly regenerated until it is within the simulation area.

For all test networks, the following simulation parameters were used: $R_i = 100$ and $V_i = 10$ for all $i \in N(t)$, $S_A = 0.75R_i$, $S_U = 0.50R_i$, $\rho = 0.10$, and $T = 1000$. Fig. 3 presents the overall procedure of the simulation to test the proposed flocking-based heuristic. Performance measure $Q(t)$ is calculated as follows:

$$Q(t) = \frac{2}{nu(nu - 1)} \sum_{i=1}^{nu} \sum_{j=i+1}^{nu} \tau(i, j, t)$$

where $\tau(i, j, t) = 1$ if there is a path between the $i$th and $j$th users at time $t$, and $\tau(i, j, t) = 0$, otherwise. In other words, performance measure $Q(t)$ represents the percent of connected user pairs at time $t$. The outcome of the simulation, $Q$, is the average connectivity of users over the mission time. This simulation procedure is used in the following sections to collect data for statistical analysis.

4.2. Statistical Analysis and Observations

A comprehensive statistical analysis was performed to investigate the effects of agent behavior parameters $w_s$, $w_c$, $w_e$, and $\alpha$ on the overall network connectivity. The test networks were simulated for all possible combinations of parameters $w_s$, $w_c$, and $w_e$={$0.0, 0.25, 0.5, 0.75, 1.0$} and $\alpha$ = {$0.5, 1.0$}. Thirty random simulation replications were performed for each combination of the parameter values. As a result, 7200 samples were collected for each test network after excluding the cases with both $w_s = 0$ and $w_c = 0$ because in these cases agent behaviors are random. The collected data for each test network was analyzed using linear regression to investigate the relationships
Randomly determine $p_i(1) \quad \forall i \in N(t)$

for $t = 1, \ldots, T$ do
\begin{itemize}
    \item Evaluate $v_i(t) \quad \forall i \in A(t)$
    \item Deploy agent $i$ to position $p_i(t+1) \quad \forall i \in A(t)$
    \item Move user $i$ to $p_i(t+1) \quad \forall i \in U(t)$
    \item Create network $G(t+1)$
    \item Evaluate $Q(G(t+1))$
\end{itemize}

}\}

Return $Q = \frac{1}{T} \sum_{t=2}^{T} Q(G(t))$

Figure 3: Procedure of the simulation.

between the agent behavior parameters and the average network connectivity as follows:

$$\hat{Q} = \beta_0 + \beta_1 w_s + \beta_2 w_c + \beta_3 w_e + \beta_4 \alpha$$

Table 1 summarizes the fitted regression coefficients for the behavior parameters. The value of a regression coefficient indicates the direction and magnitude the relationship between the corresponding agent behavior and the average network connectivity $Q$. For example, a high positive regression coefficient indicates a strong positive correlation between the related agent behavior and network connectivity. On the other hand, a negative regression coefficient indicates that the related agent behavior is detrimental to the network connectivity. If an agent behavior is not statistically significant with a confidence level of 0.05, its corresponding regression coefficient is underlined in the table. Based on the results in Table 1, the following conclusions can be inferred.

- Separation behavior had a statistically significant positive impact on the overall network connectivity. The impact of separation behavior depends on network density (the number of nodes per unit area) and the number of agents. Separation behavior became more influential with the increasing number of users and agents (i.e., increasing network density). This pattern can be clearly observed in Table 1. It is intuitive to expect that the more nodes that a MANET has, the more likely that
links will be established among its nodes. Separation behavior in dense networks encourages an even distribution of agents over the simulation area, which in turn increases the likelihood of establishing connections between disconnected network segments.

• Cohesion behavior had also a statistically significant positive impact on the overall network connectivity. However, its impact was diminished as the network got denser. In dense networks, nodes are more likely to establish and maintain their links without depending on the cohesion behavior. Therefore, the cohesion behavior was not a significant factor for the test networks with 30 users. In sparse networks, however, it is critical for agents to maintain their links because once they are separated from other nodes, it is less likely that they can establish new connections. In this study, therefore, cohesion behavior was the most significant factor to increase network connectivity for the test networks with 10 users.

• Random exploration behavior had a negative impact in the case of sparse networks and a positive impact in the case of dense networks. Exploration behavior increases mobility of agents and may cause agents to break up their existing links or to establish new ones. Clearly, this behavior was detrimental to the network connectivity in the case of sparse networks where the probability of establishing new links was small.

• In majority of the cases, a higher level of connectivity was achieved with $\alpha = 1$ than with $\alpha = 0.5$. This observation suggests that the current state of the network should be primarily considered in agent deployment decisions.

Fig. 4 illustrates the contour plots of the average network connectivity versus the different values of the behavior parameters $w_s$ and $w_c$. These plots also support the observations in the regression analysis. In summary, the separation behavior was the dominant factor in the dense networks. The highest network connectivity, which is denoted by lighter shades in Fig. 4, was achieved toward $w_s = 1.0$, and the cohesion behavior was the dominant factor in the sparse networks (the highest network connectivity was achieved toward $w_c = 1.0$). In all cases, the lowest network connectivity was observed for $w_s = 0.0$ and $w_c = 0.0$, in which agents do not exhibit
Table 1: Estimated Regression Coefficients

<table>
<thead>
<tr>
<th>Test Network</th>
<th>Regression Coefficients of Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( w_s )</td>
</tr>
<tr>
<td>(10,2)</td>
<td>0.389</td>
</tr>
<tr>
<td>(10,3)</td>
<td>0.795</td>
</tr>
<tr>
<td>(10,4)</td>
<td>1.24</td>
</tr>
<tr>
<td>(10,5)</td>
<td>1.61</td>
</tr>
<tr>
<td>(20,2)</td>
<td>0.430</td>
</tr>
<tr>
<td>(20,3)</td>
<td>0.920</td>
</tr>
<tr>
<td>(20,4)</td>
<td>1.51</td>
</tr>
<tr>
<td>(20,5)</td>
<td>1.71</td>
</tr>
<tr>
<td>(30,2)</td>
<td>0.260</td>
</tr>
<tr>
<td>(30,3)</td>
<td>0.895</td>
</tr>
<tr>
<td>(30,4)</td>
<td>1.49</td>
</tr>
<tr>
<td>(30,5)</td>
<td>2.11</td>
</tr>
</tbody>
</table>

*underline:* Statistically insignificant parameter with a confidence level of 0.05.

4.3. Comparing Performance of the Flocking-Based Heuristic

In this section, a quadratic mixed-integer programming formulation, called Problem \( ALOC(t) \), is used to benchmark the performance of the proposed flocking-based approach. The detailed description of Problem \( ALOC(t) \) is given in Appendix A. Problem \( ALOC(t) \) determines the optimal locations of agents based on the global state of the network at time \( t \). In the simulation procedure given in Fig. 3, Problem \( ALOC(t) \) is solved optimally at each time period \( t \) for the current state of the network, then the optimal locations of agents are set as their starting locations at the next time period. In other words, the locations of agents are determined optimally instead of using the proposed flocking-based heuristic during the simulation. Therefore, Problem \( ALOC(t) \) provides a good benchmark to compare the performance the proposed flocking-based heuristic. It should be noted that agent deployments separation and cohesion behaviors. Therefore, it can be concluded that the proposed flocking-based heuristic is effective in guiding agents to improve network connectivity.
determined by Problem $ALLOC(t)$ are not optimal for the entire simulation period because the optimization in each time period is considered independently. On the other hand, determining an optimal agent deployment policy for the entire simulation period is computationally intractable for the test networks studied in this paper.

Problem $ALLOC(t)$ was embedded into the simulation using the Gurobi 5.0 Java Programming Language Interface. During the simulation, a model was dynamically created based on the current state of the network at each time period, and then the created model was solved optimally by the Gurobi Mixed-Integer Quadratically Constrained Programming solver. In the simulation, the same parameters were used as given in the previous section. All simulation runs were performed in a PC with 2.4GHz Dual-Core CPU with 4 GB memory. Test networks with up to 20 users and 5 agents were used. Larger networks could not be simulated due to the CPU time requirement of the Gurobi solver. The parameters of the flocking-based heuristic were set based on the results of the previous computational experiments as follows:
$w_s = 0.5$, $w_c = 0.75$, and $w_e = 0$ for networks with 10 users; $w_s = 0.75$, $w_c = 0.5$, and $w_e = 0$ for networks with 20 users; and $\alpha = 0.9$, $S_A = 0.75R_i$, $S_U = 0.50R_i$ for all networks. Thirty random simulation replications were performed for each test network. In addition, the flocking-based heuristic was run in the pure exploration mode (i.e., $w_s = 0$, $w_c = 0$, and $w_e = 1$). In the pure exploration mode, the behavior of an agent is similar to random walk. Therefore, the pure exploration mode sets a baseline to compare the performance of the proposed flocking-based heuristic and the Gurobi Quadratically Constrained Programming (GQCP) approach. Table 2 presents the averaged $Q$ values archived by the pure exploration mode (Random), the flocking-based heuristic (Flocking), and the GQCP approach in 30 random simulation replications. In addition, the percent improvements of the flocking-based heuristic and the GQCP approach over the pure exploration mode (columns F/R and G/R in the table, respectively), as well as the percent improvement of the GQCP approach over the flocking-based heuristic are given in the table. As seen in Table 2, both the flocking-based heuristic and the GQCP approach provided significantly higher levels of user connectivity with respect to the random behavior of agents. The difference is more significant for the test networks with 10 users than the test networks with 20 users as it is easier to form random links in dense networks. The performance of the flocking-based heuristic is par with the performance of the GQCP approach. For each test network, the average user connectivity levels ($Q$ values) achieved by the flocking-based heuristic and the GQCP approach in 30 simulation replications were compared using two-sided t-test assuming equal variances. This statistical analysis has shown that the performances of these two approaches are not statistically different for most test networks, excluding test network (20,2) in which the GQCP approach provided a 8% higher connectivity level on the average. In several test networks, the flocking-based heuristic provided slightly higher levels of connectivity than the GQCP. As stated earlier, an agent deployment policy determined by the GQCP approach is not optimal for the entire simulation time because time periods are considered independently. The best deployment decision at a time period may have adverse affects for future periods. Therefore, it is possible that the flocking-based heuristic can provide higher levels of connectivity than the GQCP approach does. The computational experiments in this section have shown that the flocking-based heuristic is very promising. It should be noted that a simulation replication using the GQCP approach to determine agent locations took several hours of CPU time due to the computational burden of optimally
solving Problem $ALOC(t)$.

Table 2: Performance Comparison of the Flocking-Based Heuristic

<table>
<thead>
<tr>
<th>Test Network</th>
<th>Averaged $Q$ in 30 Simulation Replications</th>
<th>Percent Difference(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random (R)</td>
<td>Flocking (F)</td>
</tr>
<tr>
<td>(10,2)</td>
<td>14.8</td>
<td>18.2</td>
</tr>
<tr>
<td>(10,3)</td>
<td>16.4</td>
<td>21.1</td>
</tr>
<tr>
<td>(10,4)</td>
<td>17.7</td>
<td>22.2</td>
</tr>
<tr>
<td>(10,5)</td>
<td>17.5</td>
<td>23.2</td>
</tr>
<tr>
<td>(10,6)</td>
<td>18.5</td>
<td>24.5</td>
</tr>
<tr>
<td>(20,2)</td>
<td>22.3</td>
<td>23.6</td>
</tr>
<tr>
<td>(20,3)</td>
<td>24.1</td>
<td>26.2</td>
</tr>
<tr>
<td>(20,4)</td>
<td>24.9</td>
<td>28.5</td>
</tr>
<tr>
<td>(20,5)</td>
<td>26.6</td>
<td>29.7</td>
</tr>
</tbody>
</table>

$^a$ Statistically significant difference based on two-sided t-test assuming equal variances with significance level of 0.05.

5. Conclusions

This paper introduced a flocking-based heuristic to improve the network connectivity in MANET through special nodes, called agents, that are guided by flocking-based behaviors. Due to the dynamic nature of MANET, agent deployment decisions must be made on-line, in a short time, and based on incomplete information about the global state of the network. Current approaches to the problem are limited by their dependency on a centralized network management system. The proposed flocking-based heuristic has a minimum overhead and can be implemented in a decentralized and asynchronous manner. The simulation experiments have shown that the connectivity of MANET can be improved by a flocking-based heuristic. The simulation experiments have also shown that effectiveness of various agent behaviors highly depends on the density of the network. This observation suggests that an adaptive strategy to guide agent behaviors might be beneficial. Further research is needed to develop an adaptive strategy.
Appendix A. Description of Problem $ALOC(t)$

In this section, a quadratic mixed-integer programming formulation, called Problem $ALOC(t)$, is introduced. Problem $ALOC(t)$ is used to benchmark the performance of the proposed flocking-based heuristic. Problem $ALOC(t)$ determines the optimal locations of agents based on the global state of the network at time $t$. Problem $ALOC(t)$ assumes that the locations of all nodes are known, and agent deployment decisions can be communicated to all agents.

In the formulation given below, time index $t$ is omitted in the decision variables for the sake of presentation simplicity. Problem $ALOC(t)$ is a multi-commodity network flow problem where decision variable $f_{sh}$ represents the flow from user node $s$ to user node $h$ through arc $(i, j)$. Node set $N$ includes both user nodes ($U$) and agent nodes ($A$). Constraints A.1 through A.3 are node flow balance constraints. The objective function aims to maximize the total flows among user nodes. Note that variable $F_{sh}$, representing the total flow from user node $s$ to user node $h$, is bounded by one. Therefore, the objective function maximizes the number of connected user pairs.

Arc set $E$ includes two types of arcs, user arcs (i.e., $\{(i, j) : \min(R_i, R_j) \leq d_{ij}, \forall i, j \in U, i \neq j\}$) and agent arcs (i.e., $\{(i, j) : \forall i \in N, \forall j \in A, i \neq j\}$). For the purpose of optimization, all arcs to/from agents are included in the formulation although some of these arcs might be longer than the maximum communication range in the current network state. Constraints A.4 through A.8 are used to calculate Euclidian distances of agent arcs. If an agent arc has a distance longer than the maximum communication range, its capacity is set to zero by Constraint A.9. Constraints A.10 and A.11 enforce arc capacity constraints on commodity flows.

Constraints A.12 through A.15 are used to calculate the distance that each agent $i$ moves within a time period, and then Constraint A.16 makes sure agent $i$ cannot move more than maximum allowed distance $V_i$ in a time period. Constraint A.17 and A.18 are used to fix the locations of users within a time period. Note that the locations of the users and agents at the beginning of time period $t$ are input to Problem $ALOC(t)$. 

18
\[ C \text{ set of user pairs (commodities), i.e, } C = \{(s, h): s, h \in U, s < h\} \]

\[ f_{s,h}^{ij} \text{ flow of commodity (s, h) through arc (i, j) } \]

\[ F_{sh} \text{ total flow from source node } s \text{ to sink node } h \]

\[ u_{ij} \text{ flow capacity of arc (i, j) } \]

Notation:

\[ d_{ij}^x \] rectilnear distance between nodes \( i \) and \( j \) in the \( x \)-axis direction

\[ d_{ij}^y \] rectilnear distance between nodes \( i \) and \( j \) in the \( y \)-axis direction

\[ d_{ij} \] Euclidian distance between nodes \( i \) and \( j \)

\[ \Delta x_i \] Distance traveled by agent \( i \) in the \( x \)-axis direction

\[ \Delta y_i \] Distance traveled by agent \( i \) in the \( y \)-axis direction

Problem \( ALOC(t) \):

\[
\text{Maximize } z = \sum_{(s,h) \in C} F_{sh} \\
\text{st.}
\sum_{(i,j) \in E} f_{s,h}^{ij} = 0 \quad \forall (s, h) \in C, \forall i \in N \setminus \{s, h\} \quad (A.1)
\]

\[
\sum_{(s,j) \in E} f_{s,h}^{sj} - \sum_{(j,s) \in E} f_{s,h}^{js} = F_{sh} \quad \forall (s, h) \in C \quad (A.2)
\]

\[
\sum_{(h,j) \in E} f_{s,h}^{jh} - \sum_{(j,h) \in E} f_{s,h}^{j} = -F_{sh} \quad \forall (s, h) \in C \quad (A.3)
\]

\[
d_{ij}^x \geq x_i - x_j \quad \forall i \in N, j \in A, i < j \quad (A.4)
\]

\[
d_{ij}^y \geq x_j - x_i \quad \forall i \in N, j \in A, i < j \quad (A.5)
\]

\[
d_{ij}^x \geq y_i - y_j \quad \forall i \in N, j \in A, i < j \quad (A.6)
\]

\[
d_{ij}^y \geq y_j - y_i \quad \forall i \in N, j \in A, i < j \quad (A.7)
\]

\[
(d_{ij}^x)^2 + (d_{ij}^y)^2 \leq (d_{ij})^2 \quad \forall i \in N, j \in A, i < j \quad (A.8)
\]

\[
d_{i,j} - \min(R_i, R_j) \leq M(1-u_{ij}) \quad \forall i \in N, j \in A, i < j \quad (A.9)
\]

\[
f_{s,h}^{ij} \leq u_{ij} \quad \forall (s, h) \in C, i \in N, j \in A, i < j \quad (A.10)
\]

\[
f_{s,h}^{ji} \leq u_{ij} \quad \forall (s, h) \in C, i \in N, j \in A, i < j \quad (A.11)
\]

\[
\Delta x_i \geq x_i - x_i(t) \quad \forall i \in A \quad (A.12)
\]

\[
\Delta x_i \geq x_i - x_i(t) \quad \forall i \in A \quad (A.13)
\]

\[
\Delta y_i \geq y_i(t) - y_i \quad \forall i \in A \quad (A.14)
\]

\[
\Delta y_i \geq y_i(t) - y_i \quad \forall i \in A \quad (A.15)
\]

\[
(\Delta x_i)^2 + (\Delta y_i)^2 \leq (V_i)^2 \quad \forall i \in A \quad (A.16)
\]

\[
x_i = x_i(t) \quad \forall i \in U \quad (A.17)
\]
\[ y_i = y_i(t) \quad \forall i \in U \quad (A.18) \]

\[ u_{ij} \in \{0, 1\} \quad (A.19) \]

\[ x_i, y_j \geq 0 \]

\[ 0 \leq f_{ij}^{sh}, F_{sh} \leq 1 \]


