Distributed Learning Automata Based Data Dissemination in Networked Robotic Systems

Gerald Henderson and Qi Han(✉)

Department of Computer Science, Colorado School of Mines, Golden, USA
gxhenderson@alumni.mines.edu, qhan@mines.edu

Abstract. Networked robotics systems often work in collaboration to accomplish tasks. The random environments the robots work in render any previous contact data between robots useless as the contact patterns are different for each deployment. In the case of military and disaster scenarios, delivering data items quickly is imperative to the success of a mission. However, robots have limited battery and need a lightweight protocol that maximizes data delivery ratio and minimizes data delivery latency while consuming minimal energy. We present two learning automata based data dissemination protocols, LADD and sc-LADD. LADD uses learning automata with direct connections to all neighboring nodes to make efficient and accurate forwarding decisions while sc-LADD uses learning automata and exploits the clustering nature of the robotic systems to abstract clusters/groups and reduce the number of decisions available to the learning automata, which also reduces overhead.

Keywords: Data dissemination · Networked robotic systems · Learning automata

1 Introduction

In many applications of collaborative networked multi-robot systems, robots need to communicate with each other or a remote server. Data dissemination can be used to send tasks to robots. If this process is not efficient, tasks may expire before the robot can perform them. This becomes increasingly important for critical tasks. Consider a disaster scenario where robots are searching for surviving civilians. Critical updates received by an emergency response personnel coordinating the search mission may need to be disseminated to the robots quickly to save civilians in the most dangerous areas of the disaster zone. Without efficient data dissemination, an area of the disaster zone could potentially go unexplored. Furthermore, multiple robots may need to receive the data based on their capabilities.

This work is supported in part by NASA grant 80NSSC18M0048.

Y. Yin et al. (Eds.): MobiCASE 2019, LNICST 290, pp. 126–140, 2019.
https://doi.org/10.1007/978-3-030-28468-8_10

We specifically consider the scenarios where multiple robots work in groups/swarms. A swarm of robots is a collective group of robots working together to complete a task. In this paper, we use group, swarm, cluster interchangeably. Each robot in the swarm may perform the same task or may perform different parts of a singular, collaborative task. Because robots in a swarm may perform different tasks within the same mission, it is possible that only a certain percentage of the total swarms need to receive data; each of the swarms may also have a different number of nodes that need the data. These types of networks have varying or intermittent network connectivity, resulting in unreliable message transmission.

This work presents Learning Automata based Data Dissemination (LADD) and Swarm Centric Learning Automata based Data Dissemination (sc-LADD), two protocols for data dissemination that use learning automata (LA) to make decisions that are adaptive to the environment; consider residual energy, mobility, link quality, hop count, and local delivery ratio; and works in a scenario which *a-priori* data about cumulative node contacts and centrality is unavailable. Each protocol uses LA to determine the next best hop based on the current network parameters. Detection and tracking of the swarms/groups is outside the scope of this work and has been well studied [21].

2 Related Work

Our work is related to several areas of research in the literature. While this work focuses on data dissemination, it is similar to data forwarding. Data forwarding refers to sending data to one single destination, while data dissemination aims to deliver data to multiple receivers. As we consider the scenarios where multiple robots communicate wirelessly in an ac-hoc manner, essentially forming a a Delay Tolerant Network (DTN), we will focus on techniques developed for DTNs, instead of traditional wired networks. Machine Learning based approaches have been used in a Delay Tolerant Network (DTN) for data forwarding [2,5,10,13,17,19].

Data dissemination in DTNs has been addressed using graph theory approaches [3,18], community based approaches [6–8,12,15,22–24], or machine learning based approaches. Reinforcement learning techniques have been used for data dissemination to adapt to the dynamic nature of DTNs. For instance, QL has been used to dynamically adjust the broadcast rate of a node [20]. In LAFTRA [14], LA share a goodness table between nodes that helps choose the next hop node. FROMS [4] treats an action of its Q-Learning model as a set of sub-actions to determine goodness of a path. Other work use RL techniques to select best forwarding nodes by considering energy consumption [9,17].

On-Demand Multicast Routing Protocol (ODMRP) [11] is a popular routing protocol designed for wireless ad-hoc networks. It uses a mesh network to create routes on demand instead of proactively. ODMRP determines sets of forwarding nodes used to send data to multicast groups. The mesh network makes the network more stable under intermittent connectivity and helps improve delivery ratio. To avoid the delay the route acquisition incurs, the protocol sends a route discovery packet with the initial data packet. ODMRP listens to multicast addresses. This means the network must know which nodes are registered to which multicast addresses prior to being deployed. In reality, our target scenarios assume that we do not know when nodes may need to receive data or what groups nodes may be a part of during deployment. This means ODMRP will fail as a data dissemination protocol in our targeted scenarios.

3 Swarms and Learning Automata

A swarm consists of multiple robots performing some mission. While the swarm at the high level can have randomly distributed movements, each robot within the swarm moves within a maximum distance from the center of the swarm, making the movements much more predictable and reliable. This allows local data dissemination to occur more easily.

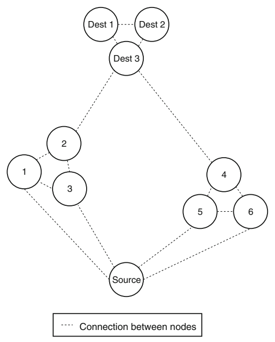
The existence of swarms can be exploited for more efficient data dissemination. At any time instant, a robot belongs to a given swarm. This allows for an abstraction when forwarding data. Once a node in the swarm receives a message, it can forward the message to the other destinations in the swarm and control local data dissemination. Instead of needing to forward to any of the respective nodes in the swarm, this work generalizes the swarm to be considered as a *supernode*.

Swarms also make it simpler to handle nodes that leave or join a swarm. Because the nodes are abstracting the swarm, a message need not be broadcast to all nodes in other swarms about the changing structure of the swarm. Finally, the swarm abstraction saves memory. Nodes do not have to consider the contacts of all nodes in a swarm, but only those in its immediate swarm as it is needed for local dissemination. This work assumes that swarms are formed as a prerequisite to the mission.

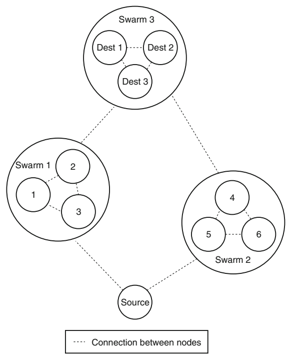
Since the swarms formed are transient (i.e. they may never meet again after the mission), there is no data about previous contacts that can be utilized for forwarding nodes. Therefore, nodes in the network must make intelligent dissemination decisions with limited network data in a random environment where data needs to be disseminated to multiple nodes. In order to extend network lifetime, the energy of nodes needs to be considered in a way that does not significantly affect latency in the network. These factors give the foundation to consider learning automata for data dissemination.

LA greatly reduces memory, overhead, and energy consumption. LA have also been shown to be effective in finding near-optimal routes. However, LA can be further improved by having an adaptive action-set where the actions available change based on the nodes that come in contact with the swarm. Furthermore, the reinforcement scheme will be able to account for energy, mobility, and proximity of nodes which will all assist in minimizing the total resources for the network. This work uses an $S-model$ LA with a variable structure. This means the goodness of an action will lie in an interval of [0, 1] and the actions available will be dynamic as nodes come into contact with other swarms.

To illustrate the potential benefits from using LA combined with swarms to assist with data dissemination, consider the example network in Fig. 1. Figure 1a shows a simple topology in which a source is attempting to deliver a data item to a set of destination nodes. To reach the destination, the source has several available forwarding options as it is connected to nodes 1, 3, 5, and 6. Figure 1b shows the same topology, but includes the use of swarms to abstract forwarding to neighboring swarms. This gives the source fewer available actions to choose from and forces swarms to handle local routing. Instead of potentially taking four different actions, each of which needing to take multiple actions to converge, the number of paths that need to be explored is reduced by half. By reducing the number of routes to explore, the LA should be able to converge more quickly and decrease delay times while maintaining a similar level of dissemination quality. Also, in this example, all three destinations are located in one swarm, one delivery to the swarm would be sufficient, thereby further reducing overhead. With the swarm abstraction, the LA will be able to adapt to the changes and re-converge more quickly when changes in optimal routes occur.



(a) An example of possible dissemination scenarios.



(b) An example of using swarms for dissemination.

Fig. 1. Benefits of using swarms in LADD

4 LADD: Distributed Learning Automata-Based Data Dissemination

We consider a network of robots as a graph $G = (V, E)$ where each robot is a vertex v_i. When two robots v_i and v_j are within communication range of each other, there is a bidirectional edge e_{ij}. We consider a single source node $s \in V$ and a set of destination nodes $D \subseteq V$. Robots are also grouped into swarms $S \subset V$, that together try to accomplish a designated mission. All the nodes inside the same swarm are within one hop of each other. Each swarm S is disjoint from all other swarms in the network such that $\forall S_i, S_j \in S \, S_i \cap S_j = \emptyset$. Each node belongs to one or no swarm at any given time such that $\forall v \in V \, v \in S_i | v \in \{\}$. A node can belong to no swarm if it is disconnected from other nodes for various reasons such as in the middle of switching its swarm or has low battery and needs to recharge. Destination nodes may be unevenly distributed among the swarms. As a mission unfolds, robots will move and destination nodes may not stay in the swarms. When data needs to be sent from source s to all nodes in D, our objective is to maximize data delivery ratio, minimize data delivery latency, and minimize overhead incurred (which includes energy consumed by the robots in message transmission and bandwidth used).

This section proposes two Learning Automata-based data dissemination schemes for solving the data dissemination problem in swarm robotic systems: LADD does not exploit swarms, sc-LADD does.

4.1 Swarm-less LADD

Nodes in LADD use the network conditions with its neighbors to adapt to the continuously changing environment instantly. Each node maintains a goodness matrix and uses the goodness values to decide where to forward its messages for the destination(s). The nodes available for forwarding will have a different goodness for each destination. The matrix allows the node's LA to make the differentiation.

Goodness Values. The goodness matrix for node i is denoted as GV_i, which is an $n \times n$ matrix where $gv[j][k]$ describes the goodness of choosing node k as i's next hop in the path to destination j. Each row of the matrix pertains to the respective node in the network if it were the destination of a message, while the column pertains to the goodness value of selecting that node on the path to the destination.

A goodness value is calculated between a node, i, and its predecessor in the path, h, based on its residual energy, current mobility, link quality with the receiver, hop count, and local delivery ratio. These networking factors give information about the goodness of paths throughout the network. It is calculated as

$$gv_{ih} = rf_i * lqf_{ih} * hcf_p * dr_i, \tag{1}$$

where rf_i is the routing factor of the current node, i, sending a response; lqf_{ih} is the link qualify factor between the current node, i, and its predecessor in the path, h; hcf_p is the hop count factor of the path, p; and dr_i is the delivery ratio of the current node, i.

- The routing factor, rf, is a node's general ability to forward. It is calculated as
$$rf_i = \frac{\text{residualEnergy}}{\text{initialEnergy}} * \left(1 - \frac{\text{velocity}}{\text{maxAllowedVelocity}}\right).$$
The first term prefers a high residual energy while the second prefers a low velocity.

- The link quality factor, lqf, is calculated as

$$lqf_{ih} = \frac{\text{rss}}{\text{minAllowedRss}}$$

where rss is the last received signal strength (RSS) between the current node and its predecessor in the path. An RSS around 0 dBm indicates a strong link quality between two nodes, while 90 dBm (the minAllowedRss) is the lowest usable signal in ad hoc networks.
- The hop count factor, hcf, is calculated as

$$hcf_p = 1 - \frac{\text{hopCount}}{\text{numberOfNodes}}.$$

Hop count is the path length from the source to the destination and can be at most the number of nodes in the network. This term will prefer paths with shorter hops.
- The local effective delivery ratio, dr, is calculated as

$$dr_i = \frac{\text{number of delivered packets}}{\text{number of sent packets}}.$$

The local effective delivery ratio is the ratio of packets sent by the node and actually delivered to its destination. Nodes that are consistently able to disseminate data to their destinations should be considered more heavily for forwarding data.

LADD Overview. Each node is equipped with an LA, goodness matrix, and enabled forwarding node set for making forwarding decisions. At the start of LADD, each element of the goodness value matrix is initialized to .5 in order to allow all paths to have a chance to be explored at some point.

Since not all nodes are in communication range of each other all the time, we use an enabled forwarding node set (i.e., enabled actions vector used in LA) which shows the list of forwarding nodes that are available to a node. As a node identifies its neighbors, the enabled forwarding node set will change. If that node then leaves the communication range of the robot, it should no longer be considered for forwarding and is removed from the enabled forwarding node set.

To select forwarding nodes, a node calculates the number of paths using local effective delivery ratio (LEDR) as follows:

$$\text{numPath} = \frac{\text{numNeighbors}}{2} * (1 - \text{LEDR}) \tag{2}$$

The local effective delivery ratio is initialized as zero, so a source node selects a max of half of its neighbors. Multiplying by this factor enables more exploration of paths when fewer packets are being delivered.

Only the first node in a path makes this calculation. Predecessors select one node for further forwarding. Once the number of forwarding nodes is decided, a node will add its *routing factor* to the cumulative routing factor of the message and its id to the path. This cumulative routing factor will tell a destination node how good of a path was chosen. The other elements of Eq. 1 cannot be utilized for this metric as they cannot be calculated until a destination is reached. When a node receives a forward message, the node must determine if it is one of the destinations. If it is, it can then calculate average routing factor of the path by dividing the cumulative routing factor by the hop count. If the average routing factor of the path is greater than the average routing factor of all data items received by the destination, it will send back a response with the calculated goodness relative to the node that sent the data item as in Eq. 1. If more destinations exist, the node forwards the data to the next hop. When a node receives a response message, it will store the received goodness value in its goodness matrix for taking that forward node as the next hop to the destination (i.e., updating gv[destination id][sender of the response's id]). It will then calculate its goodness relative to the predecessor in the path before sending the response to its predecessor.

When a node forwards a message, it decreases the goodness value of the chosen forwarding node by half in the goodness matrix (i.e., gv[destination][forward node] = gv[destination][forward node]/2).

LADD's Learning Automata. Nodes send and receive several different types of messages: *Forward*, *Response*, *Mobility*, and *Hello* messages. When receiving a *Forward* message, a node in LADD will send a *Response* down the path if it is a destination. If the average cumulative routing factor of the message is greater than or equal to its local cumulative routing factor, it will send a reward response. Otherwise, it will send a penalty response. Then, if there are more destinations, it will add its routing factor to the cumulative routing factor of the message and select a forwarding candidate for each destination (which may be a shared hop) from its neighboring nodes.

When receiving a reward *Response* message, the node will update the goodness value from the sending node in its goodness matrix. For either a penalty or reward, the node will send the response down to its predecessor in the path.

Mobility messages let neighboring nodes know when a node has begun moving more quickly. In this event, the node should be removed as an enabled action. These messages are added to a *Hello* message. When receiving a *Hello* message, a node updates or adds the neighboring node to the enabled actions vector.

Figure 2 gives a high-level overview of the data dissemination process in LADD.

4.2 Swarm-Centric LADD (sc-LADD)

Swarms create an abstraction layer for nodes in the network. In order to reduce the memory space needed by the goodness value matrix, a node only maintains its neighbors in its current swarm along with a single connection to the *supernode* of the other swarms. However, having a connection to a single node

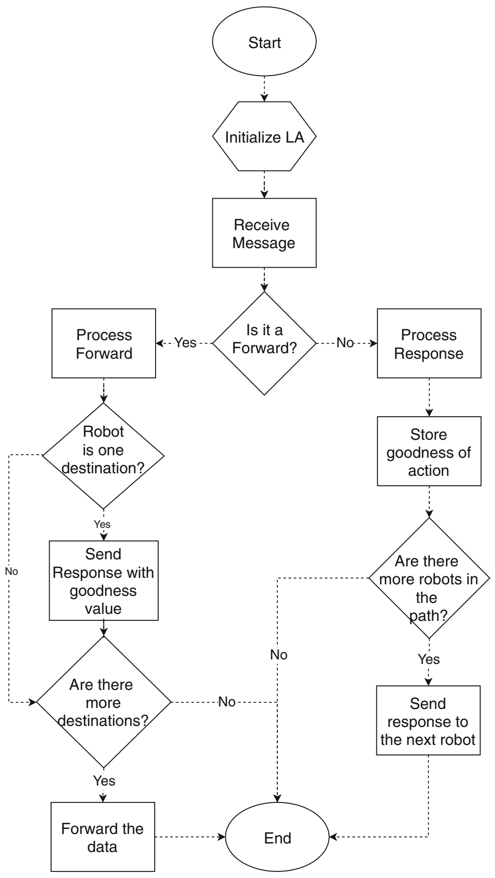

Start
Initialize LA
Receive Message
Is it a Forward?
Yes
No
Process Forward
Process Response
Robot is one destination?
Yes
No
Send Response with goodness value
Store goodness of action
Are there more robots in the path?
No
Yes
Are there more destinations?
No
Yes
Send response to the next robot
Forward the data
End

Fig. 2. A flowchart of the data dissemination process in LADD.

in the swarm may not be the most beneficial, as another node may have better routing capabilities. Therefore, sc-LADD also maintains a small table, the *swarm_supernode* table, which contains the node most suitable for forwarding in the swarm. This table maintains swarm ID, supernode reference id, and routing factor. The supernode reference id is the ID of the node currently being utilized within a swarm as the *supernode* and the routing factor is its current goodness value as a routing node. However, the routing factor is not dependent on routing to specific destinations as goodness values are, but instead its ability to route in general. Both supernode reference id and routing factor are updated as nodes in the swarm come into the communication range of a node.

As sc-LADD still applies similar concept of LA, we next discuss how this algorithm works by only describing the difference from the original LADD.

- The routing factor utilized between swarms does not consider mobility and only considers the residual energy of the node. When utilizing swarms, the granularity of decision making changes. Instead of making fine-grained decisions to each node in the network, there is more coarse-grained abstraction. Mobility of neighboring nodes is more fine-grained information, while average residual energy of the network remains coarse-grained. When considering swarms, the individual node mobility is abstracted. Removing the mobility information from the routing factor matches the more coarse-grained nature of the swarms. Therefore, routing factor for sc-LADD is calculated as:

$$rf = \frac{\text{residualEnergy}}{\text{initialEnergy}}$$

The goodness value calculation is also changed to reflect this. In addition, RSS and local effective delivery ratio are not calculated in the goodness value when using swarms. Because only energy and hop count are considered, goodness values begin to reflect the goodness of a path as opposed to an individual node. Specifically, goodness values for sc-LADD are calculated as:

$$gv = rf * hcf \tag{3}$$

– One additional change needs to occur for the cumulative routing factor stored on a node. Because routing factor now only considers the residual energy of the node, the routing factor will always be decreasing for a node. To resolve this issue, the node keeps track of the rate that it is using battery and occasionally reduces its cumulative routing factor at this same rate. This allows the cumulative routing factor to still be reflective of the good paths in the network.
– sc-LADD only maintains connections to its local swarm and the supernode from the neighboring swarms.
– sc-LADD only sets the enabled actions for a node to those of its neighboring supernodes. When a node receives a forward message, it checks if it is a destination as per the original algorithm. However, before forwarding the data items, the node will then check if any of the destinations lie in its swarm. If this is true, it will broadcast the forward message throughout the swarm. Next, it will forward the message to the neighboring swarm with the best goodness value. Before sending the message to the neighbors it has selected for forwarding, it will add all of the nodes in its swarm to the disabled actions list to ensure the swarm will not receive the message again.
– When a node has determined it is a destination, it will build the response as per the original algorithm. However, the response will only need to travel between the swarms via the supernodes that forwarded the data item. When a supernode receives a response, it will broadcast that response to the rest of the swarm before sending it back down the path to the next supernode. This ensures that all nodes have the updated goodness for routing to the swarm in the event that the supernode changes and one of the other nodes is chosen for forwarding data.

5 Performance Evaluation

Our simulation studies are conducted using NS-3 [16], a widely used network simulator. Robots communicate with each other using Wi-Fi ad hoc mode. NS-3 provides realistic simulation of Wi-Fi network conditions based on network topology and robot mobility. Using NS3 [16] and node traces generated by Bonnmotion [1], realistic scenarios were created that mimic networked robotic systems being used for disaster relief applications. Each scenario includes several parameters: number of nodes, number of swarms, node mobility, and swarm change rate. By default, we have 10 swarms, a total of 50 robots, each robot's max pause time is 60 s and max allowed velocity is 51 mph and swarms do not change. We chose 51 mph based on the specs of the DJI Phantom 4. One of these parameters is changed to determine how that parameter affects either protocol. These experiments will help evaluate the scalability and adaptability of the algorithms.

We consider several performance metrics: data delivery ratio, data delivery latency, average cost, and average energy consumption.

- Data delivery ratio is the percentage of destinations that actually received a data item.
- Data delivery latency is the average delay for all delivered destinations to receive the data items being disseminated. A low delivery latency is desired as all data should be disseminated as quickly as possible.
- Average cost is the amount of overhead produced by the data dissemination. Overhead is considered as the number of forward and response messages being sent. Reducing the amount of overhead means less energy is spent on transmitting data into the network.

Impact of Number of Nodes. Studying the impact of the number of nodes in the network gives a better indication of how having more or less nodes in the same geographical area affects the performance of our data dissemination algorithms, which provides a sense of the scalability of the algorithms.

Results shown in Fig. 3 are from what we consider the worst case scenario, where nodes constantly move up to the maximum speed of 51 mph and rarely pause, meaning that nodes and their swarms never stop in any single location for more than 60 s. The constant mobility makes it difficult for the LA to converge as it constantly needs to switch paths it just discovered. Despite these conditions, LADD is able to provide high delivery radio ranging between 96% and 98% as shown in Fig. 3a. However, sc-LADD falls about 6–10% below LADD in delivery ratio. Changes in the network propagate more quickly in LADD because of the fine-grained decision making. When considering the delivery latency shown in Fig. 3b, sc-LADD outperforms LADD in all cases but the 20-node case. However, delivery latency for both protocols decreases as more nodes are added to the network.

Finally, the overhead for sc-LADD, shown in Fig. 3c, is significantly lower than the overhead of LADD. Interestingly, after 30 nodes, sc-LADD begins to decrease in overhead while LADD's overhead only increases on a linear scale as more nodes are added. This is because the number of decisions for a node in LADD only increases, while the number of decisions for sc-LADD actually remains unchanged, causing the overhead gap to continuously grow between the two protocols.

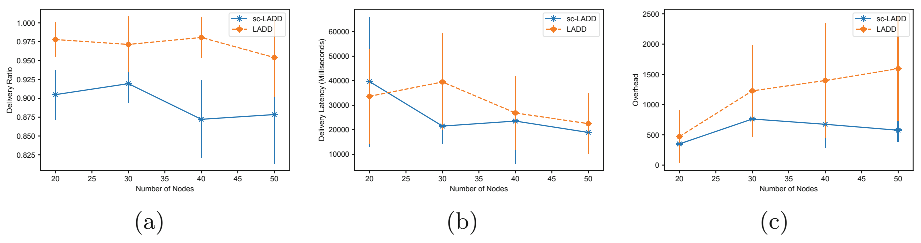


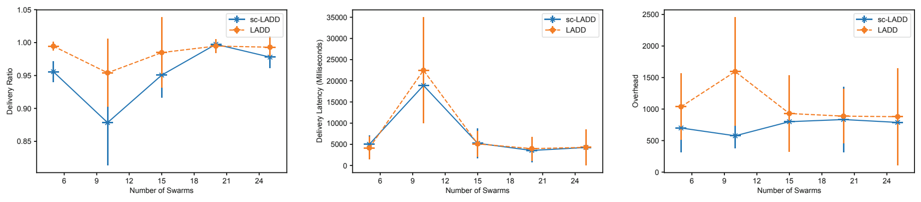
Fig. 3. Impact of number of nodes

Impact of Number of Swarms. Considering the number of swarms shows how the protocols will work with different sized swarms. Having fewer swarms indicates that there are more nodes in each swarm while having more swarms indicates there are less nodes in each swarm.

Figure 4 shows the comparison of LADD and sc-LADD with different number of swarms. As shown in Fig. 4a, delivery ratios of the two protocols are similar with LADD slightly outperforming sc-LADD; there is an exception of the 10 swarm, which is given the more extreme parameters of the number of nodes comparison. In the 20 swarm case, sc-LADD slightly outperforms LADD. Delivery latency, shown in Fig. 4b is nearly identical between the two protocols.

Finally, sc-LADD again outperforms LADD in overhead as shown in Fig. 4c. However, the overhead becomes much more similar from 15 to 25 swarms. This is most likely due to the changing sizes of the swarms. As swarms become smaller, the protocol gets closer to becoming the same as LADD and removes fewer actions for the LA to take.

Impact of Node Mobility. The max pause time simulations measure performance when nodes are able to pause from 0 to 60 min in increments of 15 min. Figure 5a shows the delivery ratio for sc-LADD and LADD are comparable with LADD outperforming sc-LADD by less than 2% in every setting. This shows how the mobility in Fig. 3 affected the results of sc-LADD. When there is more stability in the network, both protocols are able to perform well. It is important to note the slight performance decrease in the 45 and 60 min scenarios. This may be due to the methods used by Bonnmotion to create the mobility traces. While more stop time is allowed for a node, it does not guarantee that all nodes will

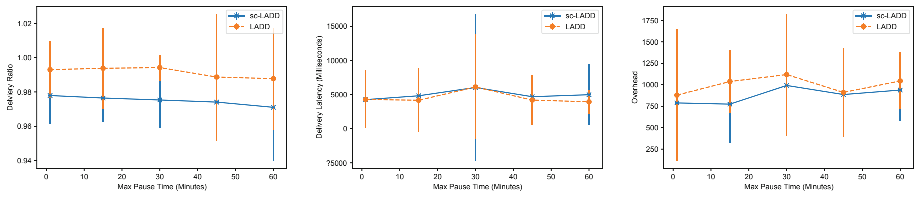


(a) (b) (c)

Fig. 4. Impact of number of swarms

stop for that time frame; in fact, nodes can still move continuously even when allowed a pause time, although this is unlikely.

Similar to other results, Fig. 5b shows that the delivery latencies of both protocols perform similarly with LADD slightly outperforming sc-LADD by at most a little over one second. Figure 5c also follows a similar pattern with sc-LADD outperforming LADD in overhead. However, the overhead is more significant in most cases than in Sect. 4 These results show that even with a similar delivery ratio, sc-LADD makes around 100–250 less transmissions per node than LADD in most cases.

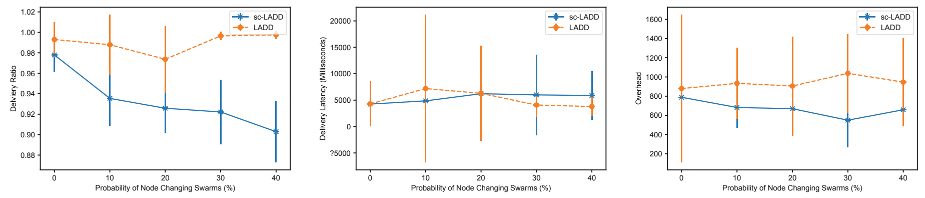


(a) (b) (c)

Fig. 5. Impact of node mobility

Impact of Swarm Change Probability. We use swarm change probability to indicate the chances of nodes switching to different swarms. With a probability of 0%, 10%, 20%, 30%, or 40%, a node will move to a different swarm in our simulation. Nodes are not leaving and then returning to the same swarm, which is a less extreme case. The 0% probability change relates to the 15 min max pause time simulation.

Figure 6a shows that while LADD still performs well, sc-LADD has a decreasing delivery ratio. This might be due to the abstraction mechanism used for sc-LADD for swarms. While it would handle a node leaving and returning to the same swarm, the mechanism is not yet suitable to handle nodes permanently changing swarms.



(a) (b) (c)

Fig. 6. Impact of swarm change probability

When a node leaves a swarm, it allows its immediate neighbors to know it is leaving and cannot receive messages. In the case of LADD, this immediate change is easily reflected as it has an immediate connection to that node and it does not matter what swarm it belongs to. Furthermore, LADD uses a fine-grained delivery mechanism to send data items directly to nodes. Any change in a node's swarm does not change another node's ability to forward data to the node changing swarms. In the case of sc-LADD, there is no direct contact with every node in a swarm. A node changing its swarm alters the path to that node. Because sc-LADD makes decisions based on paths, this is not immediately reflected in all nodes in said path. This makes it harder to forward to a node's new swarm. Future work would include updating the abstraction mechanism to alert other nodes when a node permanently changes its swarm.

Similar to the reasons stated for delivery ratio, Fig. 6b shows LADD begins to greatly outperform sc-LADD in delivery latency as nodes are more likely to switch between swarms. Nevertheless, sc-LADD still outperforms LADD in overhead. However, some of the performance benefit in the overhead can be tied to the decreasing delivery ratio.

6 Conclusion

In this work, we have created two versions of a data dissemination protocol for networked multi-robot systems that adapt to the rapidly changing topology of the mobile robotic network without the knowledge of a-priori node contact data. A learning automata based approach was taken to keep the algorithm lightweight. The simulation results indicate that the non-swarm based protocol, LADD, slightly outperforms the swarm based protocol, sc-LADD, in terms of delivery ratio in all cases. However, sc-LADD incurs significantly less overhead than LADD, while delivery latency remains similar in all scenarios. In many cases, LADD does not provide significantly higher delivery ratio than sc-LADD. This is especially true in the case of nodes pausing. In these scenarios, sc-LADD can be used to achieve a reasonable delivery ratio with much lower overhead. However, in the scenarios where LADD does provide higher delivery ratio than sc-LADD, sc-LADD could be paired with LADD as an energy conserving mode. When a node goes below a certain energy threshold, it can switch from LADD mode to sc-LADD to conserve energy. This will extend the network lifetime at the price of a lower delivery ratio.

References

1. Aschenbruck, N., Ernst, R., Gerhards-Padilla, E., Schwamborn, M.: BonnMotion: a mobility scenario generation and analysis tool. In: Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques (SIMUTools), pp. 51:1–51:10 (2010)
2. Dhurandher, S.K., Sharma, D., Woungang, I., Gupta, R., Garg, S.: GAER: genetic algorithm-based energy-efficient routing protocol for infrastructure-less opportunistic networks. J. Supercomput. **69**(3), 1183–1214 (2014)
3. Frey, H., Ingelrest, F., Simplot-Ryl, D.: Localized minimum spanning tree based multicast routing with energy-efficient guaranteed delivery in ad hoc and sensor networks. In: 2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1–8 (2008)
4. Förster, A., Murphy, A.L.: FROMS: a failure tolerant and mobility enabled multicast routing paradigm with reinforcement learning for wsns. Ad Hoc Netw. **9**(5), 940–965 (2011)
5. Gao, W., Cao, G.: User-centric data dissemination in disruption tolerant networks. In: Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), pp. 3119–3127 (2011)
6. Gao, W., Cao, G., Porta, T.L., Han, J.: On exploiting transient social contact patterns for data forwarding in delay-tolerant networks. IEEE Trans. Mob. Comput. **12**(1), 151–165 (2013)
7. Gao, W., Li, Q., Zhao, B., Cao, G.: Multicasting in delay tolerant networks: a social network perspective. In: Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), New York, NY, USA, pp. 299–308 (2009)
8. Gao, W., Li, Q., Zhao, B., Cao, G.: Social-aware multicast in disruption-tolerant networks. IEEE/ACM Trans. Netw. **20**(5), 1553–1566 (2012)
9. Hu, T., Fei, Y.: QELAR: a machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks. IEEE Trans. Mob. Comput. **9**(6), 796–809 (2010)
10. Hui, P., Crowcroft, J., Yoneki, E.: BUBBLE Rap: social-based forwarding in delay-tolerant networks. IEEE Trans. Mob. Comput. **10**(11), 1576–1589 (2011)
11. Lee, S., Gerla, M., Toh, C.: On-demand multicast routing protocol (ODMRP) for ad-hoc networks. Mob. Netw. Appl. **7** (2003)
12. Li, F., Zhao, L., Zhang, C., Gao, Z., Wang, Y.: Routing with multi-level cross-community social groups in mobile opportunistic networks. Pers. Ubiquitous Comput. **18**(2), 385–396 (2014)
13. Lindgren, A., Doria, A., Schelén, O.: Probabilistic routing in intermittently connected networks. SIGMOBILE Mob. Comput. Commun. Rev. **7**(3), 19–20 (2003)
14. Misra, S., Krishna, P.V., Bhiwal, A., Chawla, A., Wolfinger, B.E., Lee, C.: A learning automata-based fault-tolerant routing algorithm for mobile ad hoc networks. J. Supercomput. **62**(1), 4–23 (2012)
15. Nguyen, N., Dinh, T., Tokala, S., Thai, M.: Overlapping communities in dynamic networks: their detection and mobile applications. In: Proceedings of the 17th Annual International ACM Conference on Mobile Computing and Networking (MobiCom), New York, NY, USA, pp. 85–96 (2011)
16. Riley, G.F., Henderson, T.R.: The *ns-3* network simulator. In: Wehrle, K., Günes, M., Gross, J. (eds.) Modeling and Tools for Network Simulation, Chap. 2. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12331-3_2

17. Torkestani, J., Meybodi, M.: Mobility-based multicast routing algorithm for wireless mobile ad-hoc networks: a learning automata approach. Comput. Commun. **33**(6), 721–735 (2010)
18. Torkestani, J., Meybodi, M.: A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs. J. Supercomput. **59**(2), 1035–1054 (2012)
19. Wu, C., Ohzahata, S., Kato, T.: Flexible, portable, and practicable solution for routing in vanets: a fuzzy constraint q-learning approach. IEEE Trans. Veh. Technol. **62**(9), 4251–4263 (2013)
20. Wu, D., et al.: ADDSEN: adaptive data processing and dissemination for drone swarms in urban sensing. IEEE Trans. Comput. **66**(2), 183–198 (2017)
21. Yu, J., Chong, P.: A survey of clustering schemes for mobile ad hoc networks. IEEE Commun. Surv. Tutor. **7**(1), 32–48 (2005)
22. Zhang, X., Cao, G.: Efficient data forwarding in mobile social networks with diverse connectivity characteristics. In: Proceedings of IEEE 34th International Conference on Distributed Computing Systems (ICDCS), pp. 31–40 (2014)
23. Zhang, X., Cao, G.: Transient community detection and its application to data forwarding in delay tolerant networks. IEEE/ACM Trans. Netw. **25**(5), 2829–2843 (2017)
24. Zhao, W., Ammar, M., Zegura, E.: Multicasting in delay tolerant networks: semantic models and routing algorithms. In: Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM) Workshop on Delay-Tolerant Networking (WDTN), New York, NY, USA, pp. 268–275 (2005)