Journal of Internet Services and Applications

RESEARCH Open Access

EZ-AG: structure-free data aggregation in MANETs using push-assisted self-repelling random walks

V. Kulathumani[1]*, M. Nakagawa[1] and A. Arora[2]

Abstract

This paper describes EZ-AG, a structure-free protocol for duplicate insensitive data aggregation in MANETs. The key idea in EZ-AG is to introduce a token that performs a self-repelling random walk in the network and aggregates information from nodes when they are visited for the first time. A self-repelling random walk of a token on a graph is one in which at each step, the token moves to a neighbor that has been visited least often. While self-repelling random walks visit all nodes in the network much faster than plain random walks, they tend to slow down when most of the nodes are already visited. In this paper, we show that a *single step push phase at each node* can significantly speed up the aggregation and eliminate this slow down. By doing so, EZ-AG achieves aggregation in only $O(N)$ time and messages. In terms of overhead, EZ-AG outperforms existing structure-free data aggregation by a factor of at least $log(N)$ and achieves the lower bound for aggregation message overhead. We demonstrate the scalability and robustness of EZ-AG using ns-3 simulations in networks ranging from 100 to 4000 nodes under different mobility models and node speeds. We also describe a hierarchical extension for EZ-AG that can produce multi-resolution aggregates at each node using only $O(NlogN)$ messages, which is a poly-logarithmic factor improvement over existing techniques.

Keywords: Mobile ad-hoc, Random walks, Scalable and robust data aggregation, Multi-resolution synopsis

1 Introduction

The focus of this paper is on computing order and duplicate insensitive data aggregates (also referred to as ODI-synopsis) and delivering them to every node in a mobile ad-hoc network (MANET) [1–4]. We are specifically motivated by data aggregation requirements in extremely large scale mobile sensor networks [5] such as networks of UAVs, military networks, network of mobile robots and dense vehicular networks, where the number of nodes are often several thousands.

In an order and duplicate insensitive (ODI) synopsis, the same data can be aggregated multiple times but the result is unaffected. MAX, MIN and BOOLEAN OR are natural examples of such duplicate insensitive data aggregation. These queries by themselves are quite common in many applications and some examples are provided below.

As one specific example, consider the application domain of intelligent transportation systems using dense vehicular ad-hoc networks (VANETs) [6]. VANETs are mobile networks supported by both vehicle to vehicle (V2V) and vehicle to road-side infrastructure (V2I) communication, which are in turn enabled by Dedicated Short Range Communication units (DSRC) on board each vehicle [7]. VANETs can be used for improving vehicular safety as well as efficiency by dynamically updating traffic maps and providing efficient re-routes [8]. For such applications, EZ-AG can be used to generate duplicate insensitive aggregates such as the maximum speed or minimum speed in a given area (that are indicative of congestion). It can be used to answer queries such as *is there any vehicle that exceeded a certain speed?*. It can be also used to answer V2I network management queries such as *is there at least one active infrastructure unit within a given area?*. VANETs are also often augmented with environmental sensors for tasks such as pollution monitoring [9]. In such applications, aggregation queries related to the sensors can be answered using EZ-AG.

*Correspondence: vinod.kulathumani@mail.wvu.edu
[1] Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown WV 26505, USA
Full list of author information is available at the end of the article

EZ-AG can also be used for data aggregation in networks of drones, UAVs [10] and underwater robotic swarms [11]. For instance, EZ-AG can be used to answer queries such as *which is the drone with minimum or maximum battery level?* or *which robotic fish detects maximum pollution?* Aggregation queries resolved by EZ-AG can also be used for *consensus driven control applications*. For example, EZ-AG can be used to dynamically navigate networks of aerial vehicles towards the area with *minimum turbulence* [12] or to dynamically navigate a swarm of robotic fish [13] towards regions of higher vegetation.

Other duplicate sensitive statistical aggregates such as COUNT and AVERAGE can also be implemented with ODI synopsis using probabilistic techniques [4, 14]. Using these extensions, EZ-AG can be used to generate duplicate sensitive aggregates such as the number of vehicles in a road segment or average speed of vehicles in a road segment.

In static sensor networks and networks with stable links, data aggregation can be performed by routing along fixed structures such as trees or network backbones [15–18]. However, in MANETs, routing has proven to be quite challenging beyond scales of a few hundred nodes primarily because topology driven structures are unstable and are likely to incur a high communication overhead for maintenance in the presence of node mobility [19]. Therefore, structure-free techniques are more appropriate for data aggregation in MANETs. However, a simple technique like all to all flooding which involves dissemination of data from each node to every other node in the network is not scalable as it incurs an overall cost of $O\left(N^2\right)$, where N is the number of nodes in the network. Therefore, in this paper we explore the use of self-repelling random walks as a structure free method for data aggregation.

1.1 Overview of approach

Random walks are appropriate for data aggregation in mobile networks because they are inherently unaffected by node mobility. The idea is to introduce a token in the network that successively visits all nodes in the network using a random walk traversal and computes the overall aggregate. We say that a node is *visited* by a token when the node gets exclusive access to the token; the visitation period can be used by the node to add node-specific information into the token, resulting in data aggregation. Note that the concept of visiting all nodes individually differs from that of token dissemination [20, 21] over the entire network where it suffices for every node to simply hear at least one token, as opposed to getting exclusive access to a token.

Note, however that traditional random walks may be too slow in visiting all nodes in the network because they may get stuck in regions of already visited nodes. Hence, in this paper we consider self-repelling random walks [22]. A self-repelling random walk is one in which at each step the walk moves towards one of the neighbors that has been least visited [22] (with ties broken randomly). Self-repelling random walks were introduced in the 1980*s* and have been studied extensively in the physics literature. One of the striking properties of self-repelling random walks is the remarkable uniformity with which they visit nodes in a graph, i.e., without getting stuck in already visited regions.

Indeed, our results in this paper confirm that until about 85% coverage, duplicate visits are very rare with self-repelling random walks highlighting the efficiency with which a majority of nodes in the network can be visited without extra overhead. However, we observe a slow down when going towards 100% coverage because when most of the nodes are already visited, the token executing self-repelling random walk has to explore the graph to find the next unvisited node. To correct this shortcoming, we introduce a complementary *push* phase that speeds up the convergence of the random walk. The *push* phase consists of just one message from each node: before the random walk is started, each node announces its own state to all its neighbors. Note that the *push* consists of only a single hop broadcast from a node to its neighbors as opposed to a flood which consists of disseminating a node's state to all the nodes in the entire network. Thus, after the push phase, each node now carries information about all its neighbors. As a result, when the random walk executes, it does not have to visit all nodes to finish the aggregation. In fact, we show that the aggregation can finish *before* the slow down starts for the self-repelling random walk. As a result both the aggregation time and number of messages are now bounded by $O(N)$, as shown in our analysis.

1.2 Summary of contributions

- We introduce a novel structure-free technique for data aggregation in MANETs that exploits properties of self-repelling random walks and complements it with a push phase. We find that *a little push goes a long way* in speeding up aggregation and reducing message overhead. In fact, the push phase consists of just a single message from each node to its neighbors. By adding this push phase, we show that both the aggregation time and number of messages are bounded in EZ-AG by $O(N)$. In fact, we show that aggregation is completed in significantly less than N token transfers. The protocol is thus extremely simple, requires very little state maintenance (each nodes only remembers the number of times it has been visited), requires no network structures or clustering.
- We compare our results with structure-free techniques for ODI data aggregation such as gossiping and show a $log(N)$ factor improvement in

messages compared to existing gossip based techniques. We evaluate our protocol using simulations in ns-3 on networks ranging from 100 to 4000 nodes under various mobility models and node speeds. We also evaluate and compare our protocol with a prototype tree-based technique for data aggregation (i.e., structure based) and show that our protocol is better suited for MANETs and remains scalable under high mobility. In fact, *the performance of EZ-AG improves as node mobility increases.*
- Finally, we also provide an extension to EZ-AG which supplies multi-resolution aggregates to each node. In networks that are quite large, providing each node with only a single aggregate may not be sufficient. On the other hand, providing each node with information about every other node is not scalable. Hierarchical EZ-AG addresses this issue by providing each node with multiple aggregates of neighborhoods of increasing size around itself. Each node can thus have information from all parts of the network, but with a resolution that decays exponentially with distance. This idea is motivated by the fact that in many systems information about nearby regions is more relevant and important than far away regions with progressively increasing importance as distance decreases. Moreover, we also show that aggregates of nearby regions can be obtained at a progressively faster rate than farther regions. Hierarchical EZ-AG uses only $O(NlogN)$ messages and outperforms existing techniques for multi-resolution data aggregation by a factor of $log^{4.4}N$.

1.3 Outline of the paper

In Section 2, we describe related work and specifically compare our contributions with existing work in structured protocols, structure free protocols and random walks. In Section 3, we state the system model. In Section 4, we describe the EZ-AG protocol. In Section 5, we analytically characterize the bound on messages and time for EZ-AG. In Section 6, we describe a hierarchical extension for EZ-AG. In Section 7, we describe the results of our evaluation using ns-3 and compare EZ-AG with a prototype tree-based protocol for data aggregation. We conclude in Section 8.

2 Related work

2.1 Structure-based protocols

The problem of data aggregation and one-shot querying has been well studied in the context of static sensor networks. It has been shown that in-network aggregation techniques using spanning trees and network backbones are efficient and reliable solutions for the problem [15–18]. However, in the context of a mobile network, such fixed routing structures are likely to be unstable and could potentially incur a high communication overhead for maintenance [19]. In this paper, we have systematically compared EZ-AG with a prototype tree-based technique for data aggregation and have shown that it outperforms the tree-based idea in mobile networks. We notice that the improvement gets progressively more significant as the average node speed increases.

2.2 Structure-free protocols

Flooding, neighborhood gossip and spatial gossip are three structure-free techniques that can be used for data aggregation. Note that flooding data from all nodes to every other node has a messaging cost of $O\left(N^2\right)$. Alternatively, one could use multiple rounds of neighborhood gossip where in each round a node averages the current state of all its neighbors and this procedure is repeated until convergence [23, 24]. However, this method requires several iterations and has also been shown to have a communication cost and completion time of $O\left(N^2\right)$ for convergence in grids or random geometric graphs, where connectivity is based on locality [25].

In [1, 2], a spatial gossip technique is described where each node chooses another node in the network (not just neighbors) at random and gossips its state. When this is repeated $O(log^{1+\epsilon}N)$ times, all nodes in the network learn about the aggregate state. Note that this scheme requires $O(N.polylog(N))$ messages. Our random walk based protocol, EZ-AG, requires only $O(N)$ messages. Note also that while all this prior work is on static networks, we demonstrate our results on mobile ad-hoc networks.

2.3 Random walks

Random walks and their cover times (time taken to visit all nodes) have been studied extensively for different types of *static* graphs [26, 27]. In this paper, we are specifically interested in time varying graphs that are relevant in the context of mobile networks.

Self-avoiding and self-repelling random walks are variants of random walks which bias the walk towards unvisited nodes [22]. The unformity in coverage of such random walks in 2-d lattices has been pointed out in [28]. Our paper extends the analysis of self-repelling random walks presented in [28] for application in mobile ad-hoc networks that are modeled as time varying random geometric graphs. Further, we show that by complementing self-repelling random walks with a push phase, we can complete aggregation in $O(N)$ time and messages. The idea of locally biasing random walks and its impact in speeding up coverage has been pointed out in [29] for static networks. Self-repelling random walks are different than the local bias technique presented in [29]. Moreover, we show how to improve the convergence of self-repelling random walks using a complementary push-phase and demonstrate our results on mobile networks.

In a recent paper [30], we have addressed the problem of duplicate-sensitive aggregation using self-repelling random walks and in that solution we have used a gradient technique to speed up self-repelling random walks. The short temporary gradients introduced in [30] are used to *pull* the token towards unvisited nodes so that each node is visited at least once. The solution in [30] requires $O(N.log(N))$ messages. In this paper, we address duplicate insensitive aggregation and show that it can be achieved using self-repelling random walks with just $O(N)$ messages.

3 Model

3.1 Network model

We consider a mobile network of N nodes modeled as a geometric Markovian evolving graph [31]. Each node has a communication range R. We assume that the N nodes are independently and uniformly deployed over a square region of sides $\sqrt{A}$ resulting in a network density $\rho = N/A$ of the deployed nodes. Consider the region to be divided into square cells of sides $R/\sqrt{2}$. Thus the diagonal of each such cell is the communication range R. Let $R^2 > 2clog(N)/\rho$. It has been shown that there exists a constant $c > 1$ such that each such cell has $\theta(logN)$ nodes whp, i.e., the degree of each node is $\theta(logN)$ whp. Such graphs are referred to as geo-dense geometric graphs [29]. Denote $d = \theta(logN)$ as the degree of connectivity.

The objective of the protocol is to compute a duplicate insensitive aggregate of the state of nodes in a MANET. The aggregate could be initiated by any of the nodes in the MANET or by a special static node such as a base station that is connected to the rest of the nodes. The aggregate needs to be disseminated to all nodes in the network. The protocol could be invoked in a one-shot or periodic aggregation mode.

3.2 Mobility model

We consider 3 different mobility models for our evaluations.

- The first is a random direction mobility model (with reflection) [32, 33] for the nodes. This is a special case of the random walk mobility model [34]. In this mobility model, at each interval a node picks a random direction uniformly in the range $[0, 2\pi]$ and moves with a constant speed that is randomly chosen in the range $[v_l, v_h]$. At the end of each interval, a new direction and speed are calculated. If the node hits a boundary, the direction is reversed. Motion of the nodes is independent of each other. An important characteristic of this mobility model is that it preserves the uniformity of node distribution: given that at time $t = 0$ the position and orientation of users are independent and uniform, they remain uniformly distributed for all times $t > 0$ provided the users move independently of each other [31, 33].
- The second is random waypoint mobility model. Here, each mobile node randomly selects one location in the simulation area and then travels towards this destination with constant velocity chosen randomly from $[v_l, v_h]$ [34]. Upon reaching the destination, the node stops for a duration defined by the *pause time*. After this duration, it again chooses another random destination and the process is repeated. We set the pause time to 2 s between successive changes.
- The third is Gauss Markov mobility model. In this model, the velocity of mobile node is assumed to be correlated over time and modeled as a Gauss-Markov stochastic process [34]. We set the temporal dependence parameter $\alpha = 0.75$. Velocity and direction are changed every 1 s in the Gauss Markov Model.

We consider node speeds in the range of 3 to 21 m/s. For the deployment density that we have chosen, a mapping between node speed and the average link changes per node per second is listed in Table 1. This table quantifies the link instability caused by node mobility at different node speeds. As seen in Table 1, because of high network density, the network structure is rapidly changing at the speeds chosen for evaluation.

While we have chosen the above mobility models for evaluation, we expect the results to hold even under other models such as motion on a Manhattan grid (suitable for vehicular networks). The crucial aspect of mobility that we capture in our evaluations is the high rate at which links change per second which is quantified in Table 1. Our results highlight that performance of EZ-AG actually improves with higher mobility speeds.

3.3 Metrics

A key metric that we are interested in is the number of times the token is transferred to already visited nodes. We present this in the form of *exploration overhead* which

Table 1 Mapping between speed and link changes per node per second (rounded off to integer)

Size	3 m/s	9 m/s	15 m/s	21 m/s
100	1	5	7	9
200	2	6	9	12
300	2	7	10	14
500	3	8	12	16
1000	3	9	14	18
2000	4	10	16	20
4000	4	12	18	23

is defined as the ratio of the number of token transfers to the number of unique nodes whose data has been aggregated into the token. We compute exploration overhead at different stages of coverage as the random walk progresses.

Typically, random walks are evaluated in terms of their *cover times*, which is defined as the time required to visit all nodes. For a standard random walk, the notion of physical time, messages and the number of steps are all equivalent. However, for the push assisted self-repelling random walks these are somewhat different. The total number of messages required to complete the data aggregation includes the push messages, the messages involved in the self-repelling random walk and the messages involved in disseminating the result to all the nodes using a flood. Moreover, each token transfer step itself consists of announcement, token request and token transfer messages. Thus, although proportional, the number of messages is different than the number of token transfer steps. Hence we separately characterize the number of messages during empirical evaluation.

Finally we note that since we study random walks on mobile networks, the notion of time is also related to node speed. Moreover, when dealing with wireless networks, time also involves messaging delays. Therefore, during empirical evaluation we separately characterize the actual convergence time (in seconds) along with the number of steps (i.e., number of token transfers).

4 Protocol

EZ-AG consists of 4 phases as shown in Fig. 1a. These phases are described below. The steps involved in the self-repelling random walk phase are shown in Fig. 1b. The communication cost in each of these phases is analyzed in Section 5.

Aggregation request phase: The node requesting the aggregate first initiates a flood in the network to notify all nodes about the interest in the aggregate. Note that each node broadcasts this flood message exactly once. This results in N messages.

Push phase: Once a node receives this request, it *pushes* its state to its neighbors. Each node uses the data received from its neighbors to compute an aggregate of the state of all its neighbors. Note that the *push* consists of only a single hop broadcast from a node to all its neighbors. In contrast, a flood consists of disseminating a node's data to the entire network. Thus, the push phase also requires exactly N messages because each node broadcasts its data once.

Self-repelling random walk phase: Soon after the initiator sends out an aggregate request, it also initiates a token to perform a self-repelling random walk. A node that has the token broadcasts an *announce* message. Nodes that receive the announce message reply back with a token *request* message and include the number of times they have been visited by the token in this request. The node that holds the token selects the requesting node which has been visited least number of times (with ties broken randomly) and transfers the token to that node. This token transfer is repeated successively. Note that nodes which hear a token announcement schedule a token request at a random time t_r within a bounded interval, where t_r is proportional to the number of times that they have been visited. Thus nodes that have not been visited or visited fewer times send a request message earlier. When a node hears a request from a node that has been visited fewer or same number of times, it suppresses its request. Thus, the number of requests received for a token announcement remains fairly constant and irrespective of network density.

We note specifically that tokens do not grow in size when they visit successive nodes because they only carry the aggregated state. Determination of the next node to visit is done with the help of individual nodes which maintain a count of the number of times they have been

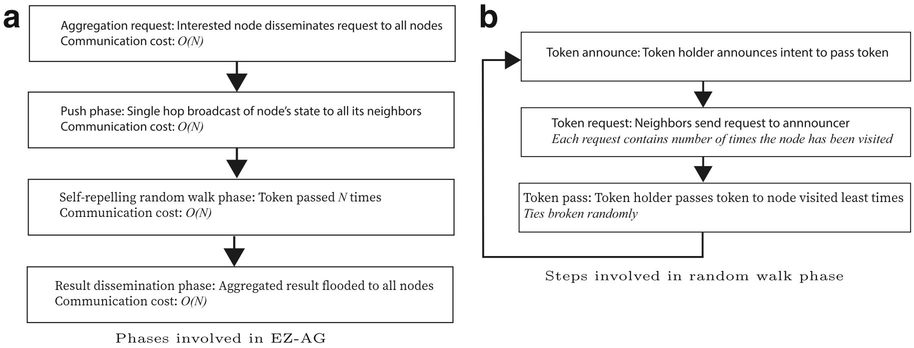

a
Aggregation request: Interested node disseminates request to all nodes
Communication cost: O(N)
Push phase: Single hop broadcast of node's state to all its neighbors
Communication cost: O(N)
Self-repelling random walk phase: Token passed N times
Communication cost: O(N)
Result dissemination phase: Aggregated result flooded to all nodes
Communication cost: O(N)
Phases involved in EZ-AG
b
Token announce: Token holder announces intent to pass token
Token request: Neighbors send request to annnouncer
Each request contains number of times the node has been visited
Token pass: Token holder passes token to node visited least times
Ties broken randomly
Steps involved in random walk phase

Fig. 1 Summary of EZ-AG protocol: EZ-AG consists of 4 phases as shown in part (**a**). The steps involved in the random walk phase are shown in part (**b**)

visited so far. This information is conveyed to the token holder after the *announce* message, which is then used to determine the next node to be visited. Thus, even at individual nodes, the state maintenance is minimal (each node only remembers the number of times it has been visited).

In the following section, we prove analytically that the aggregate can be computed from all nodes in the network whp in $O(N)$ token transfers. In the empirical evaluation, we show that the median number of token transfers is actually only kn, where $0 < k < 1$, and k is unaffected by network size. Thus, the median exploration overhead is less than 1. One can use this observation to terminate the self-repelling random walk after exactly N steps and whp one can expect that data from all the nodes has been aggregated.

Result dissemination phase: Once the aggregate has been computed, the result can simply be flooded back to all the nodes by the node that holds the result. This requires $O(N)$ messages. Another potential solution (when aggregate is only required at a base station) is to transmit the aggregated tokens using a long distance transmission link (such as cellular or satellite links) in hybrid MANETs where the *long links* are used for infrequent, high priority data.

The protocol is thus extremely simple, requires very little state maintenance, and requires no network structures or clustering.

4.1 Reliability of token transfer

The reliable transfer of tokens from one node to another is important for successful operation of EZ-AG. If a token is released by a node, but the intended recipient did not receive the token reply message, the token is lost. Reliability of token transfer can be imposed by requiring an acknowledgement from the node receiving the token and re-sending the token if an acknowledgement was not received. However, it is possible that the token was transferred correctly to a neighbor but the acknowledgement was lost or the recipient of the token moved away from the communication range of a sender. In this case, a duplicate token may be created by this process. But, since EZ-AG computes duplicate insensitive aggregates, the addition of a duplicate token will not impact the accuracy.

5 Analysis

In this section, we first show that the aggregation time and message overhead for push assisted self-repelling random walks is $O(N)$. We consider a static network for our analysis. In Section 7, we evaluate the protocol under different mobility models and verify that the results hold even in the presence of mobility.

First, we state the following claim regarding the uniformity in the distribution of visited nodes during the progression of a self-repelling random walk.

Proposition 1 *The distribution of visited nodes (and unvisited nodes) remains spatially uniform during the progression of a self-repelling random walk.*

Argument: Our claim is based on the analysis of uniformity in coverage of self-repelling random walks in [28] and in [35]. In [28], the variance in the number of visits per node of *self-repelling* random walks is shown to be tightly bounded, resulting in a uniform distribution of visited nodes across the network. More precisely, let $n_i(t,x)$ be the number of times a node i has been visited, starting from a node x. The quantity studied in [28] is the variance $(1/N)\left(\sum_i (n_i(t,x)-\mu)^2\right)$, where $\mu = (1/N)\left(\sum_i n_i(t,x)\right)$. It is seen that this variance is bounded by values less than 1 even in lattices of dimensions 2048×2048. A detailed extension of this analysis for *mobile networks* is presented in Section 7.1 which shows the uniformity with which nodes are visited during a self-repelling random walk. We use this to infer that even after the walk started, the distribution of visited nodes (and by that token, unvisited nodes) remains uniform. The result shows that the self-repelling random walk is not stuck in regions of already visited nodes - instead, it spreads towards unvisited areas.

Theorem 1 *The required number of messages for data aggregation by EZ-AG in a connected, static network of N nodes with uniform distribution of node locations is $O(N)$.*

Proof We note that the aggregation request flood and the result dissemination flood require $O(N)$ messages. During the push phase, each node broadcasts its state once and this also requires only N messages. Now, we analyze the self-repelling random walk phase.

Consider the region to be divided into square cells of sides $R/\sqrt{2}$ (see Fig. 2). Thus the diagonal of each such cell is the communication range R. Recall from our system model that each such cell has $\theta(logN)$ nodes whp at all times and there are $O(N/log(N))$ such cells. Therefore, at the end of the push phase, each node has aggregated information about its $\theta(logN)$ cell neighbors. Also note that the network can be divided into $\theta(N/log(N))$ sets of nodes that each contain information about $\theta(log(N))$ nodes within their cell. Therefore, the self-repelling random walk has to visit at least one node in each cell to finish aggregating information from all nodes.

To analyze the number of token transfers required to visit at least one node in each cell, we use the analogous coupon collector problem (also known as the double dixie cup problem) which studies the expected number of coupons to be drawn from B categories so that at least

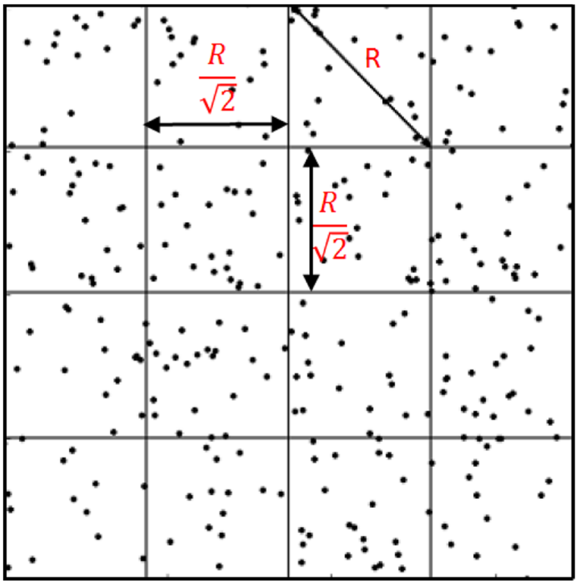


Fig. 2 Proof synopsis: Consider the region divided into square cells with diagonal size *R*. At the end of single step push phase, each node has information about all nodes in its cell. So it is sufficient for the token (performing a self-repelling random walk) to visit one node in each cell to finish aggregation

1 coupon is drawn from each category [36]. To ensure that at least 1 coupon is drawn from each category whp, the required number of draws is $O(B.log(B))$. Using this result and the fact that a self-repelling random walk traverses a network uniformly, we infer that $O((N/logN) * log(N/logN))$ token transfers are needed to visit at least 1 node in each of the $\theta(N/logN)$ cells.

Note that $log(N) > log(N/log(N))$. Hence, the required number of messages for the push assisted self-repelling random walk based aggregation protocol is $O(N/log(N) * log(N))$, i.e., $O(N)$. □

Note that in the presence of mobility, the node locations with respect to cells may not be preserved during the push phase. Therefore the generation of $\theta(N/log(N))$ identical partitions of network state as described in the above analysis may not exactly hold. However, in Section 7 we empirically ascertain that kN token transfers (where $k < 1$) are still sufficient to aggregate data from all nodes even in the presence of mobility. In fact, we observe that *the required token transfers actually decrease with increasing speed*, indicating that data aggregation using self-repelling random walks is *actually helped by mobility*.

It follows from the above result that the total time for aggregation is also $O(N)$. The impact of network effects such as collisions on the message overhead and aggregation time (if any) will be evaluated in Section 7.

In terms of communication, for data aggregation to complete, we note that each node has to at least transmit its own data once. Thus, $O(N)$ is an absolute lower bound in terms of communication messages for data aggregation. We have thus shown that EZ-AG achieves this lower bound of $O(N)$ for data aggregation and therefore is indeed quite efficient in terms of communication. Moreover, we also show that the random walk phase terminates in exactly N token passes. Also during each transfer of the token, the number of requests for the token remain fairly constant and low (See Fig. 12). Thus, it is not the case that the constants of proportionality are high either.

By way of contrast, in a pure flooding based approach, each node will have to flood the data to every other node resulting in $O\left(N^2\right)$ cost. Instead, EZ-AG first aggregates the data using $O(N)$ cost and then floods the result in $O(N)$ cost, thus resulting in a total of only $O(N)$ communication cost. The impact of this order efficiency becomes increasingly significant as network size increases.

6 Extension for hierarchical aggregation

When a network is quite large, providing each node with only a single aggregate for the entire network may not be sufficient. On the other hand, providing each node with information about every other node is not scalable. We therefore pursue an extension to EZ-AG where each node can receive multi-resolution aggregates of neighborhoods with exponentially increasing sizes around itself. This way, each node can have information from all parts of the network but with a resolution that decays exponentially with distance. This idea is motivated by the fact that in many systems information about nearby regions is more relevant and important than far away regions with progressively increasing importance as distance decreases. In this section, we describe how EZ-AG can be extended to provide such multi-resolution synopsis of nodes in a network with only $O(NlogN)$ messages.

Existing techniques for such hierarchical aggregation require $O\left(Nlog^{5.4}N\right)$ messages [1]. Thus, EZ-AG offers a poly-logarithmic factor improvement in terms of number of messages for hierarchical aggregation. Moreover, EZ-AG can also be used to generate hierarchical aggregates that are distance-sensitive in refresh rate, where aggregates of nearby regions are supplied at a faster rate than farther neighborhoods.

6.1 Description

We divide the network into square cells at different levels (0, 1, .. P) of exponentially increasing sizes (shown in Fig. 3). At the lowest level (level 0), each cell is of sides $R/\sqrt{2}$. Recall from our system model that each such cell has $\theta(log(N))$ nodes whp. For simplicity, let us denote $\theta(log(N)$ by the symbol δ. Thus, there are N/δ cells at level 0. Note that 4 adjoining cells of level i constitute a cell of

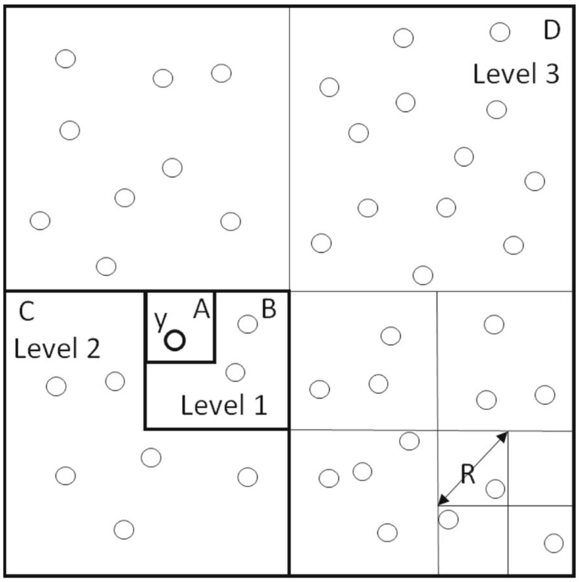


Fig. 3 Extension of EZ-AG to deliver multi-resolution aggregates: The network is partitioned into cells of increasing hierarchy where the cell at smallest level is of diagonal *R*. The node *y* shown in the figure would receive an aggregate corresponding to one cell at each level that it belongs to. In this case, it would receive aggregates for cells A, B, C and D. The largest cell *D* consists of the entire network

level $i+1$. Thus, each cell at level j has $\delta 4^j$ nodes whp. At the highest level P, there is only one cell with all the N nodes. Note that $P = log_4(N/\delta)$. At any given time, a node belongs to one cell at each level.

To deliver multi-resolution aggregates, we introduce a token and execute EZ-AG at each cell at every level. A token for a given cell is only transferred to nodes within that cell and floods its aggregate to nodes within that cell. Thus, there are N/δ instances of EZ-AG at level 0 and each instance computes aggregates for δ nodes, i.e., $\theta(logN)$ nodes.

The computation and dissemination of aggregates by different instances of EZ-AG are not synchronized. Thus, a node may receive aggregates of different levels at different times. Also, since the nodes are mobile, an aggregate at level l received by a node at any given time corresponds to the cell of the same level l in which it resides at that instant.

6.2 Analysis

Theorem 2 *An ODI aggregate at level j can be computed using hierarchical EZ-AG in* $O\left(4^j\delta\right)$ *time and messages.*

Proof Note that each cell at level j contains $\theta\left(4^j\delta\right)$ nodes whp. Therefore, using Theorem 1, EZ-AG only requires $O\left(4^j\delta\right)$ time and messages to compute aggregate within the cell. □

We note from the above theorem that aggregates at level 0 can be published every $O(\delta)$ time, aggregates at level 1 can be published every $O(4\delta)$ time and so on. Thus, aggregates for cells at smaller levels can be published exponentially faster than those for larger cells. Thus, if the tokens repeatedly compute an aggregate and disseminate within their respective cells, EZ-AG can generate hierarchical aggregates that are distance-sensitive in refresh rate, where aggregates of nearby regions are supplied at a faster rate than farther neighborhoods.

Theorem 3 *Hierarchical EZ-AG can compute an ODI aggregate for all cells at all levels using* $O(NlogN)$ *messages.*

Proof Note that a cell at level 0 contains δ nodes and there are N/δ such cells. The aggregate for cells at level 0 can be computed using $O(\delta)$ messages.

In general, there are $N/4^j\delta$ cells at level j and aggregates for these cells can be computed using $O\left(4^j\delta\right)$ messages. Summing up from levels 0 to P, the total aggregation message cost (M) for hierarchical EZ-AG can be computed as follows.

$$\begin{aligned} M &= \sum_{j=0}^{P} 4^j\delta\frac{N}{4^j\delta} \\ &= \sum_{j=0}^{P} N \\ &= O(NlogN) \end{aligned}$$

Thus, hierarchical EZ-AG can compute an ODI aggregate for all cells at all levels using $O(NlogN)$ messages. □

6.3 Comparison of hierarchical EZ-AG with gossip techniques

In [1, 2], a spatial gossip technique is described where each node chooses another node in the network (not just neighbors) at random and gossips its state. When this is repeated $O(log^{1+\epsilon}(N))$ times (where $\epsilon > 1$), all nodes in the network learn about the aggregate state. Note that this scheme requires $O(N.polylog(N))$ messages. EZ-AG requires only $O(N)$ messages.

In [1], an extension to the spatial gossip technique is described which provides a multi-resolution synopsis of the network state at each node. The technique described in [1] requires $O\left(Nlog^{5.4}(N)\right)$ messages. The hierarchical extension of EZ-AG only requires $O(NlogN)$ messages.

7 Performance evaluation

In this section, we systematically evaluate the performance of EZ-AG using simulations in ns-3. We set up MANETs ranging from 100 to 4000 nodes using the network model described in Section 3. Nodes are deployed uniformly in the network with a deployment area and communication range such that $R^2 = 4log(N)/\rho$. Thus, the network is geo-dense with $c = 2$, i.e., each node has on average $2log(N)$ neighbors whp and the network is connected whp. We test such networks in our simulations with the following mobility models: 2-d random walk, random waypoint and Gauss-Markov (described in Section 3). The average node speeds range from 3 to 21 m/s. We also consider static networks as a special case.

First, we analyze the convergence characteristics of the push-assisted self-repelling random walk phase in EZ-AG and compare that with self-repelling random walks and plain random walks. Next, we analyze the total messages and time taken by EZ-AG. Finally, we compare EZ-AG with a prototype tree based protocol and with gossip based techniques.

7.1 Coverage uniformity

First, in Fig. 4a, b and c, we show the number of times each node is visited when the self-repelling random walk has finished visiting 50% of the nodes, 75% of the nodes and 85% of the nodes. We observe that most of the nodes are just visited once and this result holds even at 1000 nodes. These graphs *highlight the uniformity with which nodes are visited as self-repelling random walks progress*. The self-repelling random walk is not stuck in regions of already visited nodes - instead, it spreads towards unvisited areas. Otherwise, one would have observed more duplicate visits to the previously visited nodes.In Fig. 4d, we analyze the distribution of number of visits at each node when 100% coverage is attained. Here, we see that most nodes are visited 2 or 3 times and the distribution falls off rapidly after that.

We then compare the uniformity in coverage with that of pure random walks. In Fig. 5, we plot the number of visits to each node until all nodes are visited at least once for a 500 node network. In comparison with self-repelling random walks (Fig. 5b), we observe that the tail of the distribution is much longer and the number of duplicate visits is much higher for pure random walks.

7.2 Convergence characteristics

Next, in Fig. 6, we show the exploration overhead of self-repelling random walk during different stages of coverage. As seen in Fig. 6, until about 85% coverage, self-repelling random walks have an exploration overhead of around 1 (irrespective of network size) but then the overhead starts to rise sharply. This is because, until this point self-repelling enables a token to find an unvisited node directly and there are very few wasted explorations. A slowdown for self-repelling random walk is noticed after this point. As a result, the exploration overhead at 100% coverage is close to 2 and moreover it increases with network size. This is what we aim to address using EZ-AG.

The exploration overhead at 100% coverage is shown in Fig. 7 for self-repelling random walks and EZ-AG (i.e., push-assisted self-repelling random walks). As seen in the figure, the exploration overhead for self-repelling random

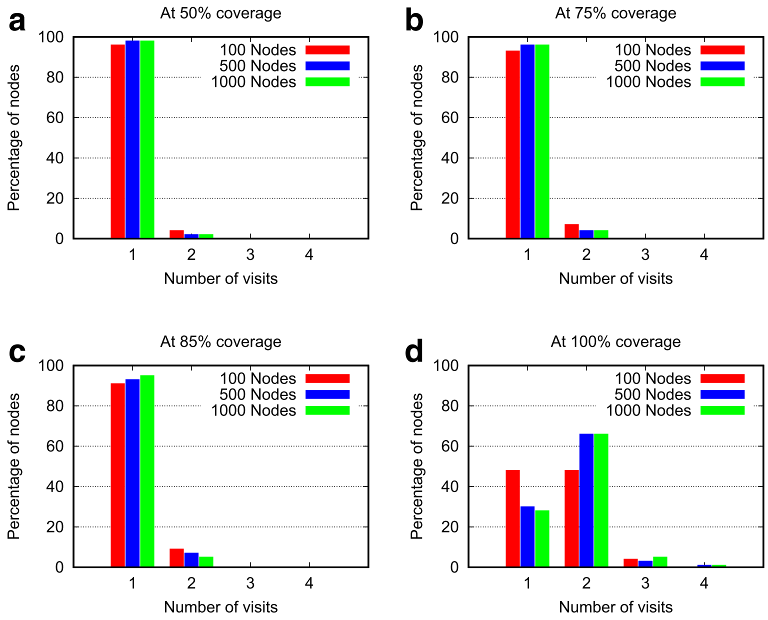


Fig. 4 Distribution of number of visits at each node at different stages of exploration of a self-repelling random walk (network size 100,400 and 1000 nodes). **a** 50% coverage, **b** 75% coverage, **c** 85% coverage, **d** 100% coverage

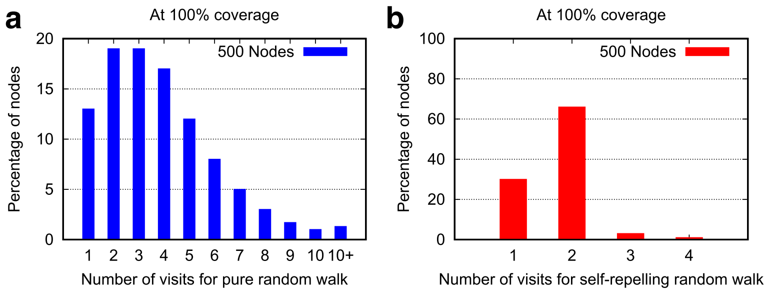


Fig. 5 Analysis of coverage uniformity: Distribution of number of visits at each node in a pure random walk and self-repelling random walk. **a** Visits distribution: Pure random walk, **b** Visits distribution: Self-repelling walk

walks grows with a logarithmic trend due to the wasted explorations towards the tail end of the random walk phase when most of the nodes are already visited. The push assisted self-repelling random walks remove these wasted explorations and as a result the median exploration overhead stays constant at all network sizes and is actually less than 1 (approximately 0.75 as seen in Fig. 7).

7.3 Impact of mobility and speed

In Fig. 8a and b, we evaluate the impact of mobility model and network speed on the exploration overhead of push assisted self-repelling random walks. We observe that even though random waypoint and Gauss Markov models do not preserve the uniform distribution of node locations, the exploration overhead exhibits a similar trend. As seen in Table 1, the network structure is rapidly changing at the speeds chosen for evaluation. Despite this, in Fig. 8b, we observe that the exploration overhead actually starts decreasing with node speed (this is shown more clearly in Fig. 9 for networks with different sizes).

7.4 Variance and terminating condition

In Fig. 10, we show the variation in exploration overhead for EZ-AG over 50 different trials at different network sizes. We observe that irrespective of network size, for 97.5% of the trials, the exploration overhead is smaller than 1. We can use this to design a terminating condition for the random walk phase of the protocol. For example, we could terminate the random walk phase after exactly N steps, and then start the dissemination of the aggregate.

7.5 Messages and time

In Fig. 11a and b, we show the total number of messages and the total aggregation time as a function of network size for the aggregation protocol based on push-assisted self-repelling random walks. The total number

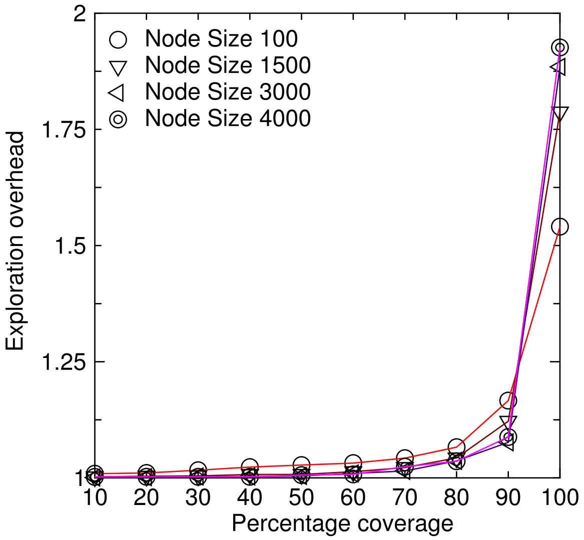


Fig. 6 Exploration overhead as a function of percentage of nodes visits for self-repelling random walks

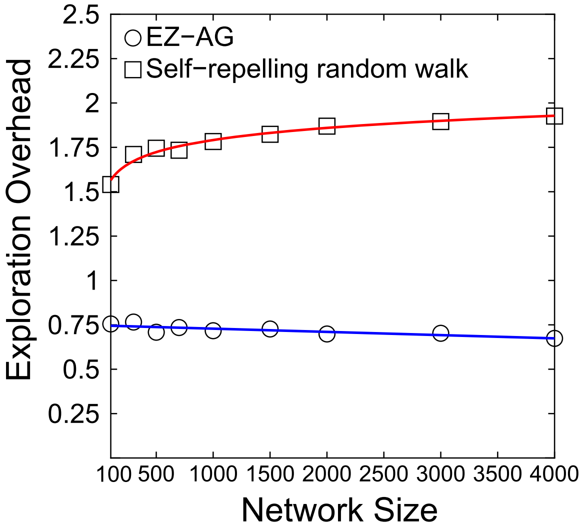
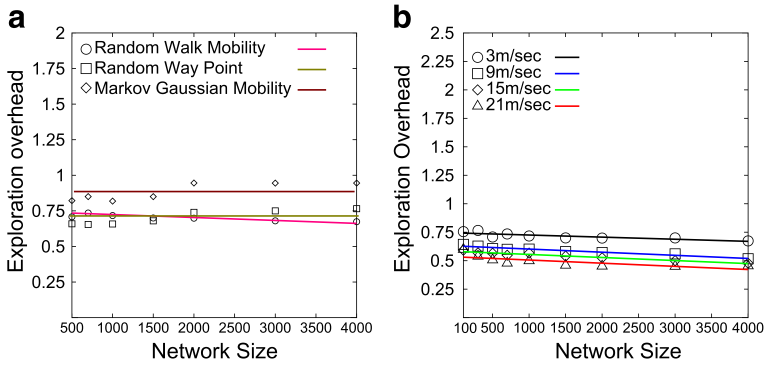


Fig. 7 Exploration overhead at 100% coverage as a function of network size



Impact of mobility models

Impact of node speed

Fig. 8 Analysis of exploration overhead of EZ-AG for different mobility models and network speeds. **a** Impact of mobility models, **b** Impact of node speed

of messages required to complete the data aggregation includes the push messages, the messages involved in the self-repelling random walk phase and the messages involved in disseminating the result to all the nodes using a flood. Note that, each token transfer step itself consists of announcement, token request and token transfer messages. These are all included in Fig. 11a which shows that the messages grow linearly with network size.

An interesting aspect of the token transfer procedure is the number of requests generated for a token during each iteration. Note that the average number of neighbors increases as $\theta(logN)$ when the network size increases. However, from Fig. 12, the number of token requests per transfer is seen to be independent of the number of neighbors. From the box plot of Fig. 12, we observe that the average number of token requests in each trial is in the range of $1 - 3$. This is because nodes that are visited less often send a request earlier than those that are visited more times. And, if a node hears a request from a node that has been visited less often than itself, it suppresses its request. Thus, irrespective of the neighborhood density, the number of token requests per node stay constant.

As seen in Fig. 11b, the total aggregation time also exhibits a linear trend. Note that the measurement of time is quite implementation specific and incorporates messaging latency in the wireless network. For instance, in our implementation each transaction (i.e., each iteration of token announcement, token requests and token passing) took on average 25 ms. But this number could be much smaller using methods such as [37] that use collaborative communication for estimating neighborhood sizes that satisfy given predicates.

7.6 Comparison with structured tree based protocol

In this section, we compare the performance of our protocol with a structured approach for one-shot duplicate

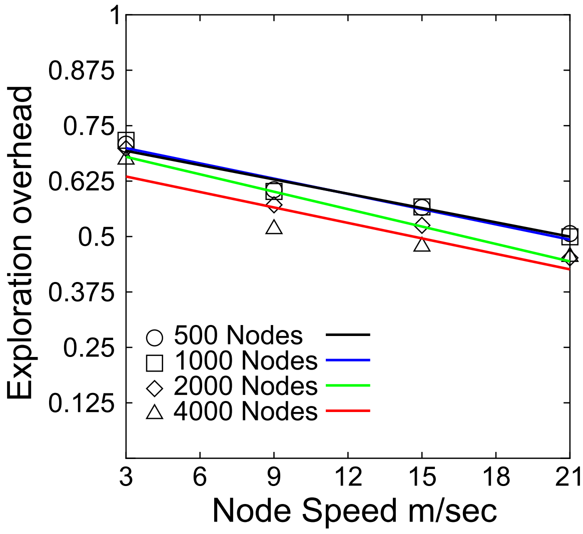


Fig. 9 Exploration overhead as a function of node speed (network size = 500 nodes)

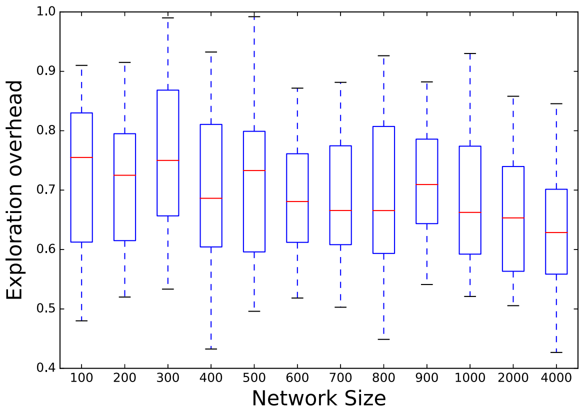
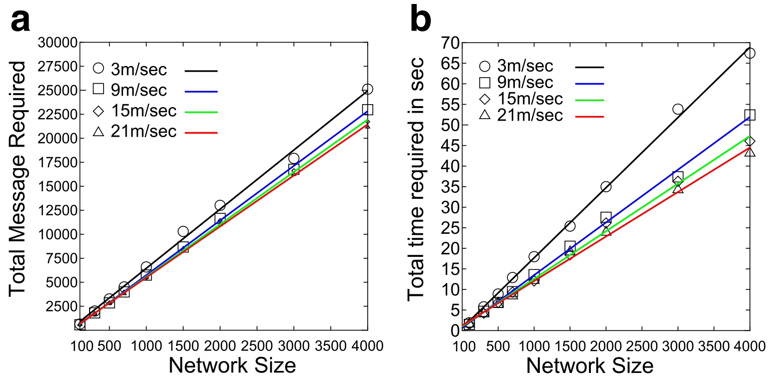


Fig. 10 Variance in exploration overhead



Total messages

Aggregation time

Fig. 11 Analysis of time and messages for EZ-AG. **a** Total messages, **b** Aggregation time

insensitive data aggregation that involves maintaining network structures such as spanning trees. For our comparison, we use a prototype tree-based protocol that we describe briefly. The idea is very similar to other tree-based aggregation protocols developed for static sensor networks [16, 17], but the key difference is that the tree is periodically refreshed to handle mobility as described below.

The initiating node maintains a tree structure rooted at itself by flooding a request message in the network. Each node maintains a *parent* variable. When a node hears a flood message for the first time, it marks the sending node as its *parent*. It then schedules a data transmission for its parent at a random time chosen within the next 25 ms. The message is successively forwarded through the tree structure until it reaches the root. During this process, a node also opportunistically aggregates multiple messages in its transmission queue before forwarding data to its parent. A message could be lost because a node's parent has moved away or due to collisions. To handle message losses, a node repeats its data transmission to its parent until an acknowledgement is received from its parent. While this basic protocol is sufficient for a static network, the network structure is constantly evolving in a mobile network. Hence, the initiating node periodically refreshes the tree by broadcasting a new request every 2 s (with a monotonically increasing sequence number to allow nodes to reset their parents). The refreshing of the tree is stopped when data from all nodes has been received at the initiating node.

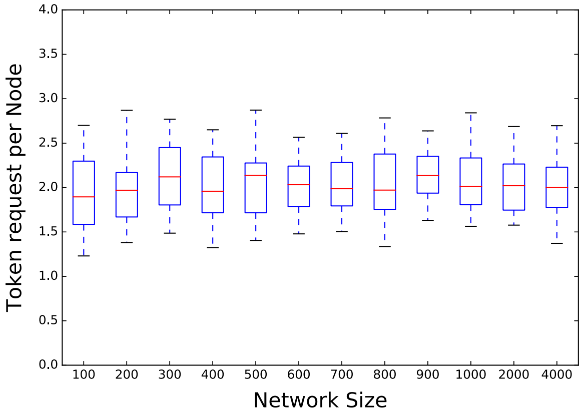


Fig. 12 Number of token requests generated per token transfer

In Fig. 13a, we compare the total messages required for the tree-based protocol and the random walk based protocol at different node speeds. As seen in this figure, for static networks the tree based protocol is more efficient. However as the mobility increases, the random walk based protocol starts increasing in efficiency. In Fig. 13b we compare the total aggregation time which also exhibits a similar trend.

In Fig. 14 we compare the total number messages as a function of network size at an average speed of 9 m/s. Here we observe that the self-repelling random walk based protocol exhibits a linear trend while the tree based protocol exhibits a super-linear trend. This is due to the potentially large number of re-transmissions experienced by the tree-based protocol in a mobile network. This graph also shows that EZ-AG is far more scalable with network size under mobility than structure-based techniques for data aggregation.

8 Conclusions

In this paper, we have presented a scalable, robust and lightweight protocol for duplicate insensitive data aggregation in MANETs that exploits the simplicity and efficiency of self-repelling random walks. We showed that by complementing self-repelling random walks with a

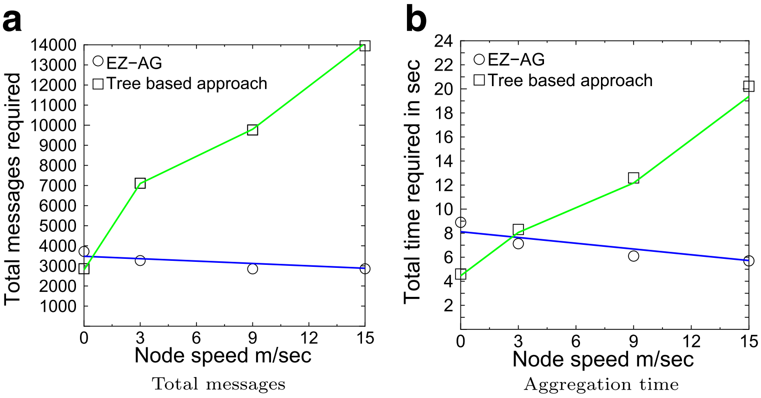


Fig. 13 Comparison of time and messages for EZ-AG and tree-based protocol at different node speeds with a network size of 500 nodes. **a** Total messages, **b** Aggregation time

single step push phase, our protocol can achieve data aggregation in $O(N)$ time and messages. In terms of message overhead, our protocol outperforms existing structure free gossip protocols by a factor of $log(N)$. We quantified the performance of our protocol using ns-3 simulations under different network sizes and mobility models. We also showed that our protocol outperforms structure based protocols in mobile networks and the improvement gets increasingly significant as average node speed increases.

We have shown that EZ-AG meets the lower bound of $O(N)$ in terms of communication requirements for aggregation. Also, each node only needs to store the number of times it has been visited. Thus, EZ-AG is lightweight in terms of both communication requirements and memory utilization. It also makes rather minimal assumptions of the underlying network. In particular, it does not assume knowledge of node addresses or locations, require a neighborhood discovery service or network topology information, or depend upon any particular routing or transport protocols such as TCP/IP.

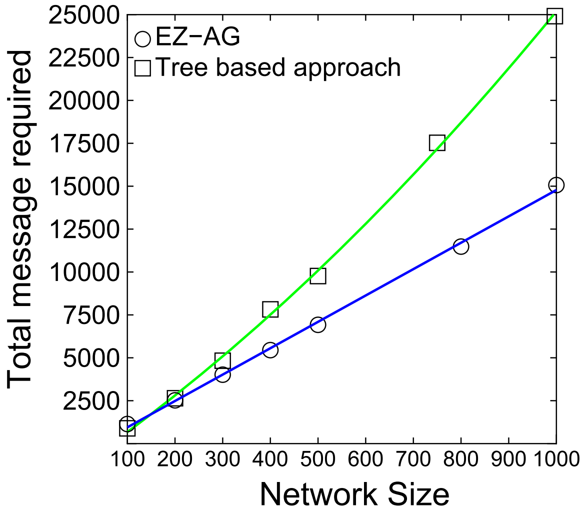


Fig. 14 Total messages as a function of network size for EZ-AG and tree based protocol (node speed = 9 m/s)

We also described a hierarchical extension to EZ-AG that provides multi-resolution aggregates of the network state to each node. It outperforms existing technique by a factor of $O\left(log^{4.4}N\right)$ in terms of number of messages.

Note that EZ-AG uses only a single step push phase, i.e. a one hop broadcast from every node to its neighbors. Extending the push phase beyond a single hop may improve the speed of convergence, but at increased complexity. Requiring each neighbor to further push the data (i.e., a 2 hop push) essentially increases the communication cost by a factor equal to the degree of connectivity d. Pushing across the network diameter is essentially flooding with a cost of $O\left(N^2\right)$. A single step push, on the other hand, maintains the communication cost at $O(N)$, while significantly speeding up the aggregation.

Funding
This research was not supported by any external funding source.

Authors' contributions

VK conceived the idea of introducing a push phase to speed up self=repelling random walks and worked on the analytic proofs. MN designed the algorithm and carried out experimental evaluations. AA contributed in the design and troubleshooting of the idea and also helped draft the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown WV 26505, USA. [2]The Samraksh Company, Dublin OH 43017, USA.

Received: 26 September 2017 Accepted: 12 January 2018
Published online: 15 March 2018

References

1. Sarkar R, Zhu X, Gao J. Hierarchical spatial gossip for multiresolution representations in sensor networks. In: International Conference on Information Processing in Sensor Networks. New York: ACM; 2007. p. 311–319.
2. Kempe D, Kleinberg JM, Demers AJ. Spatial gossip and resource location protocols. In: ACM Symposium on Theory of Computing. New York: ACM; 2001. p. 163–172.
3. Kulathumani V, Arora A. Distance sensitive snapshots in wireless sensor networks. In: Principles of Distributed Systems (OPODIS), vol. 4878. New York: Springer; 2007. p. 143–158.
4. Nath S, Gibbons P, Seshan S, Anderson Z. Synopsis diffusion for robust aggregation in sensor networks. In: Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems, SenSys '04. New York: ACM; 2004. p. 250–262.
5. Wang Y. Mobile sensor networks: System hardware and dispatch software. ACM Comput Surv. 2014;47(1):12:1–12:36.
6. Dietzel S, Petit J, Kargl F, Scheuermann B. In-network aggregation for vehicular ad hoc networks. IEEE Commun Surv Tutor. 2014;16(4):1909–32.
7. Tahmasbi-Sarvestani A, Fallah YP, Kulathumani V. Network-aware double-layer distance-dependent broadcast protocol for vanets. IEEE Trans Veh Technol. 2015;64(12):5536–46.
8. Kim G, Ong YS, Cheong T, Tan PS. Solving the dynamic vehicle routing problem under traffic congestion. IEEE Trans Intell Transp Syst. 2016;17(8): 2367–80.
9. Hu S, Wang Y, Huang C, Tseng Y. Measuring air quality in city areas by vehicular wireless sensor networks. J Syst Softw. 2011;84(11):2005–12.
10. Purohit A, Sun Z, Zhangi P. Sugarmap: Location-less coverage for micro-aerial sensing swarms. In: Proceedings of the 12th International Conference on Information Processing in Sensor Networks, IPSN '13. New York: ACM; 2013. p. 253–64.
11. Tan X. Autonomous robotic fish as mobile sensor platforms: Challenges and potential solutions. Mar Technol Soc J. 2011;45(4):31–40.
12. Kothari M, Postlethwaite I, Gu D. Uav path following in windy urban environments. J Intell Robot Syst. 2014;74(1):1013–28.
13. Yu H, Shen A, Peng L. A new autonomous underwater robotic fish designed for water quality monitoring. In: 2012 Proceedings of International Conference on Modelling, Identification and Control. Piscataway: IEEE; 2012. p. 561–6.
14. Considine J, Li F, Kollios G, Byers J. Approximate aggregation techniques for sensor databases. In: Proceedings of the 20th International Conference on Data Engineering, ICDE '04. Piscataway: IEEE; 2004.
15. Naik V, Arora A, Sinha P, Zhang H. Sprinkler: A Reliable and Energy Efficient Data Dissemination Service for Extreme Scale Wireless Networks of Embedded Devices. IEEE Trans Mob Comput. 2007;6(7):777–89.
16. Madden S, Franklin JM, Hellerstein J, Hong W. TAG: A Tiny AGgregation Service for Ad-hoc Sensor Networks. SIGOPS Oper Syst Rev. 2002;36(SI): 131–46.
17. Gnawali O, Fonseca R, Jamieson K, Moss D, Levis P. Collection tree protocol. In: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, SenSys '09. New York: ACM; 2009. p. 1–14.
18. Intanogonwiwat C, Govindan R, Estrin D, Heidemann J, Silva F. Directed diffusion for wireless sensor networking. IEEE Trans Netw. 2003;11(1):2–16.
19. Kulathumani V, Arora A, Sridharan M, Parker K, Lemon B. On the repair time scaling wall for manets. IEEE Commun Lett. 2016;PP(99):1–4.
20. Chen Y, Shakkottai S, Andrews J. On the role of mobility on multi-message gossip. IEEE Trans Inf Theory. 2013;56(12):3953–70.
21. Levis P, Patel N, Shenker S, Culler D. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In: USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI). Berkeley: USENIX; 2004. p. 15–28.
22. Byrnes C, Guttman AJ. On self-repelling random walks. J Phys A Math Gen. 1984;17(17):3335–42.
23. Friedman R, Gavidia D, Rodrigues L, Viana A, Voulgaris S. Gossiping on manets: The beauty and the beast. SIGOPS Oper Syst Rev. 2007;41(5): 67–74.
24. Boyd S, Ghosh A, Prabhakar B, Shah D. Randomized gossip algorithms. IEEE Trans Info Theory. 2006;52(6):2508–30.
25. Rabbat MG. On spatial gossip algorithms for average consensus. In: 2007 IEEE/SP 14th Workshop on Statistical Signal Processing. Los Alamitos: IEEE Computer Society; 2007. p. 705–9.
26. Lovascz L. Random walks on graphs: A survey. Combinatorics, Paul Erdos 80. 1993.
27. Ercal G, Avin C. On the cover time of random geometric graphs. Autom Lang Program. 2005;3580(1):677–89.
28. Feund H, Grassberger P. How a random walk covers a finite lattice. Physica. 1993;A(192):465–70.
29. Avin C, Krishnamachari B. The power of choice in random walks: An empirical study. In: Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems, MSWiM '06. New York: ACM; 2006. p. 219–28.
30. Kulathumani V, Arora A, Sridharan M, Parker K, Nakagawa M. Census: fast, scalable and robust data aggregation in manets. Springer Wirel Netw. 20171–18. Published online Feb 2017. https://doi.org/10.1007/s11276-017-1452-y.
31. Clementi A, Monti A, Pasquale F, Silvestri R. Information spreading in stationary markovian evolving graphs. IEEE Trans Parallel Distrib Syst. 2011;22(9):1425–32.
32. Le Boudec JY, Vojnovic M. Perfect simulation and stationarity of a class of mobility models. In: IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 4. New York: IEEE; 2005. p. 2743–54.
33. Nain P, Towsley D, Liu B, Liu Z. Properties of random direction models. In: IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 3. 2005. p. 1897–907.
34. Camp T, Boleng J, Davies V. A survey of mobility models for ad hoc network research. Wirel Commun Mob Comput (WCMC): Special Issue Mob Ad-hoc Netw. 2002;2:483–502.
35. Kulathumani V, Nakagawa M, Arora A. Coverage characteristics of self-repelling random walks in mobile ad-hoc networks. https://arxiv.org/pdf/1708.07049.pdf. Accessed Dec 2017.
36. Newman DJ, Shepp L. The double dixie cup problem. Am Math Mon. 1960;67(1):58–61.
37. Zeng W, Arora A, Srinivasan K. Low power counting via collaborative wireless communications. In: Proceedings of the 12th International Conference on Information Processing in Sensor Networks, IPSN '13. New York: ACM; 2013. p. 43–54.