REDUCING POWER AND DELAY IN WIRELESS SENSOR NETWORKS USING JUMPING ANT ROUTING ALGORITHM AND SPRAY AND FOCUS ALGORITHM

A.SINDHUJA  K.T.MATHUMITHAA

Department of Computer Science Engineering
Dr. MAHALINGAM COLLEGE OF ENGINEERING AND TECHNOLOGY

Abstract — Wireless sensor Networks (WSNs), is one of the most rapidly growing scientific domain. This is because of the development of advanced sensor nodes with extremely low cost One of the characteristic feature of WSNs compared to the traditional wireless communication networks, is the power awareness, due to the fact that the batteries of the sensor nodes have restricted lifetime and are difficult to be replaced. This is why we focus on “power awareness”. Due to its working environment and the mobility of sensor node, this kind of sensor network is very much essential to reduce power utility. We propose a jumping ant routing algorithm (JARA) which combines the advantages of reactive and proactive routing to speed up the route discovery time and reduce the route discovery overhead in sensor network, thereby reducing power. JARA, a population based algorithm provides, natural and intrinsic way of exploration of search space in optimization settings in determining optimal data aggregation. The simulation results shows improvement in energy efficiency depends on number of source nodes in sensor network which is 45% energy efficiency using optimal aggregation compared to approximate aggregation schemes in moderate number of sources. To reduce “delay” in such networks we introduce replication or “spraying” methods that can reduce the overhead of flooding-based schemes by distributing a small number of copies to only a few relays, whereas 20% energy efficiency in large number of source nodes. To route messages efficiently in such networks, we propose a scheme that also distributes a small number of copies to few relays. However, each relay can then forward its copy, instead of naively waiting to deliver it to the destination itself. This scheme exploits all the advantages of controlled replication, and could deliver the message faster thereby reducing delay. Simulation results for traditional mobility models, as well as for a more realistic “community-based” model, indicate that our scheme can reduce the delay 20 times compared to existing techniques.

Keywords—Wireless sensor networks, JARA (Jumping Ant Routing Algorithm), Spray and Focus Routing, cluster head, Cluster nodes.

I.INTRODUCTION

A Wireless Sensor Network consists of a group of spatially distributed sensor nodes which are interconnected without using wires. Each of the distributed senor nodes typically consists of one or more sensing elements, a data processing unit, communicating components and a power source,
which is usually a battery. The sensed data is collected, processed and then routed to the desired end user through a designated sink point, referred as base station. Now it has become feasible to construct multifunctional sensor nodes with advanced capabilities. Such sensor nodes are relatively of smaller size, lower cost and lesser power consumption. WSNs are originally motivated for the use in military applications, such as border monitoring. Now a days it is mainly focused on civilian applications such as environment monitoring, object tracking and bio-medical applications. A wireless sensor network operates with limited computational and sensing capabilities capable of sensing, computing and wirelessly communicating [1]. Since the sensor nodes have irreplaceable, batteries with limited power capacity, it is essential that the network be energy efficient in order to maximize the life span of the network. Large number of sensor nodes have to be networked together, direct transmissions from any specified node to a distant base station is not used, as sensor nodes that are farther away from the base station will have their power sources drained much faster than those nodes that are closer to the base station. On the other hand, minimum energy multi-hop routing scheme will result in rapidly drain energy resources of the nodes, since these nodes engage in the forwarding of a large number of data messages (on behalf of other nodes) to the base station.

**Ant Routing Algorithm:**

Ant Routing Algorithms are inspired by the behavior of real ant colonies. Many studies have discussed the use of this algorithm to solve various problems [1],[8],[10]. Since it is reliable, survivable and dynamic, the optimum solution for this algorithm is determined by creating artificial ants. The artificial ants search the solution space as real ants search their environment for food. The probabilistic movement of ants in the system allows the ants to study new paths and to re-explore old visited paths. The strength of the pheromone deposit directs the artificial ants towards the best paths, while the pheromone evaporation lets the system forget old information and avoid quick convergence to sub-optimal solutions. The probabilistic selection of the paths enables searching large numbers of solutions. Ant routing is a self-configured, self-built protocol, which can reduce the number of broadcast messages that need to be sent and which maintains several multi paths. When a node wishes to find and maintain a path to its destination, it sends forward ants searching for this destination. A forward ant moves in the network searching for the destination using the intermediate nodes’ probability routing tables and the local heuristic information. Forward ants collect information about paths and intermediate nodes local information as they travel along the path. When a forward ant reaches its destination, the information carried by this forward ant is graded. The forward ant is then killed, and a backward ant is generated.

**Related Work**

WSNs consist of hundreds of even thousands of sensor nodes which may be sparsely distributed in remote locations. A typical sensor node consumes most of its energy during communication. However, energy expenditure takes place while performing sensing and data processing too. Hence each and every protocol should be so designed, that minimum energy should be consumed during sensing, processing and communication. The application of a aggregation approach helps reduce the amount of information that needs to be transmitted by performing data fusion at the aggregate points before forwarding the data to the end user.
The backward ant carries its corresponding forward ant’s grade and the identities of the intermediate nodes in the path. The backward ant is sent back along the reverse path of its corresponding forward ant. As backward ants move in the reverse path, the intermediate nodes modify their pheromone table based on the path grade carried by the backward ant, and accordingly update their pheromone table probability. Finally, the source node receives the backward ants, updates its tables and kills the backward ant.

**Zone routing protocol:**

Proactive and reactive protocols both have particular flaws. The Zone Routing Protocol (ZRP) combines the advantages of both into a hybrid scheme, utilizing a proactive mechanism to discover a node’s local neighborhood, and applying a reactive protocol to communicate between neighborhoods. As mentioned earlier, the ZRP provides a framework for the routing protocols. The separation of nodes local neighborhood from the global topology of the entire network allows the application of different methods, thus exploiting each technique’s features in a given situation. These local neighborhoods are called zones. Each node may be within multiple overlapping zones, and each zone may be of a different size. The “size” of a zone is not determined by geographical measurement, as might be expected, but instead is given by a radius of length q, where q denotes the number of hops to the perimeter of the zone.

II.JUMPING ANT ROUTING ALGORITHM

This investigation presents a jumping ant routing algorithm (JARA) that combines the advantages of both Ant Routing Algorithm and Zone Routing Protocol, while also employing jumping mode to reduce the pro-active overhead. The algorithm is discussed in two parts. The first part relates to how each node uses proactive routing protocol to maintain the topology of q hops. The intra-network is assumed to have been already established by one of proactive routing protocols, and thus to have already generated an intra-network table. The other part concerns how each node applies ant routing to discover paths outside its zone. Each node has its own zone, and each ant obtains a route within q hops. Hence, each ant jumping q hops distance is considered as one movement. This work explains and simulates the proposed algorithm, using q = 2. The setting q = 2 was chosen because it is sufficient to demonstrate the predominance of the proposed algorithm. Each node in our algorithm can discover detailed information of neighboring nodes within q hops. These neighboring nodes can be organized into a zone called the intra-network. Those nodes within a zone are classified into boundary and interior nodes.

Fig.1: Zone of node A with q = 2.

The minimum distance of a boundary node minimum distance to the central node is exactly equal to the zone radius q. Nodes with minimum distances of less than q are called interior nodes. The central node in the figure is node A. Nodes B, D, E, H, F and J denote the boundary nodes; nodes C, G and I are interior nodes, and node K and L are outside the routing zone. Each ant in node A adopts the pheromone table to choose a boundary
node as the next node. If the ant specifies node B, then it must move to node B via node C. Node C only relays packets from the central node A to a boundary node. Hence, the JARA can speed up the route discovery and find a better path. The subsection explains route discovery, then discusses the effect of changes in network topology.

A. Route discovery:

Ants are classified as forward, backward and guide ants. Forward and backward ants are responsible for collecting path information and updating pheromone. A guide ant constructs an optimal path when all the backward ants have arrived at source a node, or when the network topology has changed. Every node also has a pheromone table.

The movement of route discovery is as follows:

1. The source node creates several forward ants to search for destination. The ants gather path information as they travel along the path.
2. A node creates a backward ant when a forward ant arrives there.
3. The backward ants are sent back following the Reverse path, and update the pheromone table.
4. The guide ant is generated when all backward Ants have arrived at the source. It updates the routing table along the optimized path, and constructs an optimal path.

B. Forward ant:

Every node in the network can be considered as a source, destination or intermediate node. A node that wants to find a path to a destination sends forward ants to search for this destination and obtain path information. When a forward ant is generated by source node, it adopts the pheromone table to obtain the next visiting node and record the path information. According to the routing principle, the next visiting node of an ant depends only on the probability in the pheromone table. In the proposed algorithm, ants prefer to move to a node that has not been visited. Such behavior is introduced to prevent ants from being enticed into the same route, thus losing the advantage of exploration. The values of the probability are calculated according to local heuristic information. A forward ant moving to an intermediates node utilizes the probability of pheromone table, adds the next destination node to intermediates node stack, and obtains local heuristic information to update path information of the forward ant packet. If the forward ant moves to the interior node, then the interior node need not do anything, but only relays the forward ant to the next destination node. The forward ant is killed when it arrives at the destination node, and a backward ant is then created. The destination node also employs path information to obtain a grade to assign to the backward ant.

C. Backward ant:

When the backward ant is received, if this node is intermediate node of the backward ant’s stack, then the node collects the grade from the backward ant’s stack and then updates the pheromone table using the grade of the backward ant, and sends the ant to the next intermediate node. If the node is an internal node, then the node looks up the routing table and transmits the ant to the next destination. The backward ant is killed when it arrives at the source node. The pheromone updating function increases the pheromone value on the incoming link, and decreases the values on other links, using the following function:

For destination D, at intermediate node i,
Where \( f(q) \) is the evaporation function, and \( g(q) \) denotes the enforcement function. The aim of the evaporation function is to help the system forget the old information quickly. A higher value of \( q \) implies a faster evaporation. Generally, the evaporation function should be given a small value. The enforcement function helps the system increase the amount of pheromone on the edges. When the grade \( q \) increases, the enforcement function should be increased. This section considers why the guide ant is adopted, and begins by considering how this algorithm indicates where to deliver packets without the guide ant. Either of the following two methods described below. The first method is to determine the routing when the packet has already flowed on networks, and the other method is to utilize the backward ant. Using the first approach, the route discovery is processed earlier. Determining the routing while packets are transmitted on networks is time-consuming, and may degrade cause the performance of transmission. Routing depends on each backward ant when using the second method. Such a method increases the frequency of the routing update, may even cause records to be sent along an unnecessary path. The guide ant can carry data, so does not create additional packets. The operations are described in detail later. The guide ant also repairs broken routes to a destination. The source node generates a guide ant to construct an optimal path. The guide ant updates the routing table, and can determine whether the resident node is a boundary or interior node. If the node is a boundary node, then the guide ant updates the routing table by referring to the pheromone table and then choosing the maximum probability node in the pheromone table. Otherwise, if the node is an interior node, then the guide ant updates the routing table and the node acts simply as a forwarder.

### III. PROPOSED SOLUTION

**Network topology changed:**

A guide ant is generated when the network topology changes. The guide ant can carry data and choose a new optimal path. This section discusses two cases, where the changed node is an interior node, when it is a boundary node.

**Changed node is interior node:**

When an interior node is (removed or deleted), a new path is built using the intra-network table, and is adopted by the guide ant to update the routing table. Therefore, route discovery does not need to be performed again. The guide ant is killed when it arrives at a node in the original path, restoring the general packet transmission. Fig. 2, illustrates an example of this case. Node A has a path to node C via node B, and node B is the interior node of node A. When node B is broken, node A finds another node to node C using “looking for intra-network table”.

In Fig. 3, node A discovers a new path to node C via node D, and therefore generates a guide ant to guide this path. The guide ant updates the routing table in nodes.
Fig. 3: Discovering a new path for moving an interior node.

In a heterogeneous environment, things are not that simple. Some nodes may be “better” relays for a given destination. Such, for example, could be nodes that tend to see the destination more often (e.g. work in the same building, or in general belong to the same social network [9, 13])

Fig. 4: Removal of boundary node.

D. Data Aggregation:

The application model considered for this work consists of a single destination (base station) and multiple sources. Since the nodes are wirelessly connected which communicates to neighbors in vicinity, therefore multi-hop communication is used to reach the destination. It is assumed that density of nodes gives a connected node graph. For an application setting, data aggregation is applied in the network. “Data aggregation merges message data in-network while traversing through network” it is also termed as data fusion. The aggregation gain can be measured as (original – aggregated) original in the given application message size. The aggregation suffers from delay termed as aggregation delay. There is a tradeoff in delay and gain in aggregation. The simulation study reveals that energy-efficiency is related to number of source nodes in correlated sensing.

IV. SPRAY AND FOCUS

Existing spraying schemes [16, 17], generate and distribute (“spray”) a small, fixed number of copies or “forwarding tokens” to a number of distinct relays. Then, each relay carries its copy until it encounters the destination or until the TTL (time-to-live) for the packet expires. By having multiple relays looking independently and in parallel for the destination, these protocols create enough diversity to explore the sparse connectivity graph more efficiently, and can discover a short path-over-time to the destination. Although such schemes have been shown to perform well in some scenarios [16, 17], they require a high amount of mobility by network nodes to achieve this performance. However, in many practical situations, the mobility of each node is limited to a
small local area for the majority of time. An example where such local mobility might arise could be, for example, that of a university campus, where most people tend to stay or move locally within their buildings for long stretches of time [11]. To make our point more clear, consider for example the “Spray and Wait” scheme [16, 17] in such a scenario. This scheme consists of two phases: in the first phase it distributes a fixed number of copies to the first few relays encountered, and in the second phase each of these relays waits until it encounters the destination itself (i.e. “Direct Transmission” [19]). It is easy to see that, here, this scheme would spread all its copies quickly to the node’s immediate neighborhood, but then few if any of the nodes carrying a copy might ever see the destination [3]. What is more, if the network is not too sparse, there might exist partial paths over which a message copy could be transmitted fast to a node closer to the destination. Yet, in schemes like Spray and Wait a relay with a copy will naively wait until it moves within range of the destination itself. This problem could be solved if a sophisticated single copy scheme is used to further route a copy after it’s handed over to a relay, a scheme that takes advantage of transmissions (unlike Direct Transmission). With this in mind, we propose Spray and Focus, which in the second phase (“focus” phase) rather than waiting for the destination to be encountered, each relay can forward its copy to a potentially more appropriate relay, using a carefully designed utility based scheme. In the next few sections, we describe our protocol in detail.

E. Spraying Phase:

When a new message gets generated at a source, and needs to be routed to a given destination, Spray and Focus first enters the “Spray phase” for this message. When a new message is generated at a source node it also creates L “forwarding tokens” for this message. A forwarding token implies that the node that owns it, can spawn and forward an additional copy of the given message, according to the following rules:

- each node maintains a “summary vector” with IDs of all messages that it has stored, and for which it acts as a relay; whenever two nodes encounter each other, they exchange their vectors and check which messages, they have in common (as in epidemic routing).
- if a node (either the source or a relay) carrying a message copy and n > 1 forwarding tokens encounters a node with no copy of the message, it spawns and forwards a copy of that message to the 2nd node; it also hands over \( \frac{n}{2} \) forwarding tokens and keeps \( \frac{n}{2} \) for itself; (Binary Spraying [17])
- when the node has a message copy but only one Forwarding token for this message, then it can only forward this message further according to the rules of the “Focus phase”.

F. Spraying mechanism:

Another interesting question is how the number of copies (or forwarding tokens) should be distributed to different relays. In a homogeneous environment (i.e. IID node movement) it is beneficial to spread messages as quickly as possible, as all nodes are statistically equivalent [1]. What matters mainly there is how many relays are looking in parallel. It is proven in [17] that, if node movement is IID and nodes forward copies only to new nodes, the algorithm that minimizes spraying time is Binary Spraying [17]. In a heterogeneous environment, things are not that simple. Some nodes may be “better” relays for a given destination. Such, for example, could be nodes that tend to see the destination more often (e.g. work in the same building, or in general...
belong to the same social network [9, 13]). Ideally, we would like to be able to choose as relays the L nodes that most frequently encounter the destination. In an offline version of the problem, we could potentially formulate a stochastic version of the linear program described in [6] and solve for the minimum expected delivery time. However, in the online version of the problem, waiting for a “better” relay incurs a cost, because it means that opportunities to spread extra copies are forfeited. In other words, there are two conflicting strategies that can potentially reduce delay when a new node is encountered,

(1) Spawn and forward an extra copy right-away to increase parallelism, and
(ii) Defer forwarding until a relay that is much more Correlated” with the destination is encountered

G. Node encounters:

First, when we refer to a “node encounter”, we assume that nodes periodically transmit beacons to recognize each other’s presence. We expect that the period of this beacon would have some effect on the performance of our protocol (if beacons are not sent often enough, some forwarding opportunities might be missed [3]). However, we assume that this is an issue that is handled by the underlying media access (MAC) protocol [5], and that, ideally, nodes “encounter” each other as soon as they come within communication range. Furthermore, there is an overhead involved in the exchange of the message summary. However, each message’s ID is expected to occupy only to have faster transfer, thereby reducing the delay. These node encounters are essentially needed for reducing the delay.

H. Number of copies used:

Another interesting question is how many copies to use per message, i.e. what the value of L should be. In general, we would like this number to be only a small percentage of the total number of nodes. For the case of simple controlled replication, or Spray and Wait, analytical expressions exist that can calculate the number of copies needed to achieve an average delay that is α times the optimal one [17].

I. Focus Phase:

When a relay for a given message has only one forwarding token left for that message, it switches to the “Focus phase”. Unlike Spray and Wait, where in the Wait phase messages are routed using Direct Transmission (i.e. forwarded only to their destination) [16, 17], in the Focus phase a message can be forwarded to a different relay according to a given forwarding criterion. Specifically, these forwarding decisions are taken based on a set of timers that record the time since two nodes last saw each other. Although the use of last encounter timers has been proposed in the past (e.g. [4]), we argue that any scheme that takes advantage of these timers needs to be carefully designed for the specific environment in hand (e.g. sparse networks with stochastic mobility) in order to achieve good performance. Let us turn then our attention to these timers. (Due to lack of space we omit some of the details of our utility-based mechanism.

J. Age of last encounter timers with transitivity:

A number of different utility functions could be envisioned for this purpose. These could also take into account other relevant information (e.g. GPS position, speed, history of encounters, etc.) in addition to the timer values. However, it is beyond the scope of this paper to evaluate all these
options, and we defer this for future work. Some efforts towards the design of multi-parameter utility functions can be found in [12]. Here, for simplicity, we will assume that these timers is our utility function (i.e. messages get forwarded to nodes with smaller and smaller timer values for the destination). A gradient-based scheme can then be used to maximize the utility function for the destination. We summarize here the functionality of the Spray and Focus protocol. Each node maintains a vector with IDs of all messages that it has stored, and for which it acts as a relay; whenever two nodes encounter each other, they exchange their vectors and check which messages they have in common; each message also carries a TTL (time-to-live). When a new message is generated at a source node create $L$ “forwarding tokens”, with $L$ chosen, if a node (either the source or a relay) carries a message copy and (i)$n > 1$ forwarding tokens-perform Binary Spraying. (ii) $n = 1$ forwarding token - perform Utility-based Forwarding according to the last encounter timers used as the utility function. Hence by using this technique the delay can be reduced to a maximum. Although node mobility is still IID (uncorrelated), each node now takes a very long time to move from one side of the network to another, and thus carrying a message is not as beneficial as in the Random Waypoint case. Figure.6 depicts the number of transmissions and the average delay of all schemes for this scenario.

![Figure.6: Random Walk Mobility.](image)

Here, the few copies are spread locally, and then each message relay takes a very long time to traverse the network and reach the destination. Even if the number of copies were increased, the delay of the spraying phase would still dominate performance, since new nodes are found very slowly. On the other hand, Spray and Focus can overcome these shortcomings and excel (unless the network is too sparse), achieving the smallest delay with only a few extra transmissions. Note also that, despite using the same utility function as Spray and Focus, Utility Flooding is still plagued by its flooding nature. This problem was even more pronounced when other existing utility functions were used [10]. This implies that in disconnected networks, the use of a utility function is not enough by itself to improve performance, but rather has to be combined with controlled replication.

### V.IMPLEMENTATION & RESULTS:

The Jumping Ant Routing Algorithm is simulated in MATLAB with a setting of sensor network of 50 nodes. The neighborhood is obtained from the random topology. Set of source nodes 20, 30&40 and a destination is considered for generating optimal routing tree using this algorithm. On varying the routing parameters and weights of nodes, shortest distance and correlation, the optimal routing tree in sensor network is obtained. The fig.7 below shows the convergence of optimal solution globally in the setup after 1000 iterations.
VI. CONCLUSION

A Jumping ant routing based solution for the Optimal routing Problem has been implemented and investigated. Extensive simulation is carried out for correctness of algorithm. It is observed that routing energy efficiency depends on the number of sources. The results of simulation reveal that this optimal routing algorithm save energy up to 45% for moderate number of source nodes. The proposed Algorithm shortens the route discovery time and reduces the route discovery overhead, especially in dense topologies. The simulation results shows that, using spray and focus, the delay can be reduced to 20 times compared to the existing techniques. Hence this “dual” mechanism improves energy efficiency and minimizes delay.

VII. REFERENCES
