An ant swarm-inspired energy-aware routing protocol for wireless ad-hoc networks

Sudip Misraa, Sanjay K. Dhurandherb, Mohammad S. Obaidatc,* Pushkar Guptab, Karan Verma, Prayag Narulab

a School of Information Technology, Indian Institute of Technology, Kharagpur, India
b CAITFS, Division of Information Technology, Netaji Subhas Institute of Technology, University of Delhi, New Delhi, India
c Department of Computer Science, Monmouth University, NJ, USA

ARTICLE INFO

Article history:
Received 4 January 2010
Received in revised form 19 June 2010
Accepted 19 June 2010
Available online 25 June 2010

Keywords:
Ad-hoc networks
Energy-efficiency
Routing
Ant colony optimization
Simulation
Performance evaluation

ABSTRACT

Primitive routing protocols for ad-hoc networks are “power hungry” and can therefore consume considerable amount of the limited amount of battery power resident in the nodes. Thus, routing in ad-hoc networks is very much energy-constrained. Continuous drainage of energy degrades battery performance as well. If a battery is allowed to intermittently remain in an idle state, it recovers some of its lost charge due to the charge recovery effect, which, in turn, results in prolonged battery life.

In this paper, we use the ideas of naturally occurring ants’ foraging behavior (Dorigo and Stuetzle, 2004) [1] and based on those ideas, we design an energy-aware routing protocol, which not only incorporates the effect of power consumption in routing a packet, but also exploits the multi-path transmission properties of ant swarms and, hence, increases the battery life of a node. The efficiency of the protocol with respect to some of the existing ones has been established through simulations. It has been observed that the energy consumed in the network, the energy per packet in the case of EAAR are 60% less compared to MMBCR and the packets lost is only around 12% of what we have in AODV, in mobility scenarios.

1. Introduction

Extending battery life of nodes is a critical issue in most ad-hoc networks. Conventional routing algorithms such as AODV [2], DSR [3] and TORAd, [4] ignore the residual battery of nodes. Data transmission between two nodes is typically done in these protocols through the shortest path routes. The algorithms instrumental in route finding in such protocols often result in faster depletion of battery in the nodes that are incident on the heavily used routes in the network (for example, the ones which lie in the shortest path routes).

The identification of the above-mentioned limitations with using conventional routing protocols for ad-hoc networks is not novel to our work. Quite a few energy-aware/power-aware schemes (e.g. [5–11]) exist in the literature. Let us discuss only a few notable ones below.

The Minimum Total Transmission Power Routing (MTPR) scheme [5] minimizes the total transmission power consumption of nodes in a route. This algorithm calculates the total power needed to transmit the packet over the route by adding transmission power required to transmit, at each node. It then chooses the path which uses minimum power. But because MTPR fails to consider the remaining power of nodes, it does not always necessarily succeed in extending the lifetime of each host.

Singh et al. proposed the Min-Max Battery Cost Routing (MMBCR) [6] scheme, which considers the residual battery power capacity of nodes as the metric in order to extend the lifetime of nodes. It calculates the battery power for each node in all the paths that could be discovered. It uses the battery power of weakest node (node having minimum battery left within a path) to check how long will the path last. Like when the weakest node goes off, path will not transfer due to break in the path. This way MMBCR tries to choose a path whose weakest node has the maximum remaining power amongst the weakest nodes in other possible routes to the same destination. However, MMBCR does not guarantee that the total transmission power is minimized over a chosen route. The Conditional Max-Min Battery Capacity Routing (CMMBCR) scheme [7] considers both the total transmission energy consumption of routes and the remaining power of nodes. MTPR is used when all the nodes forming a path (note that one path is sufficient) have remaining battery capacity that is above a so-called battery protection threshold (, and MMBCR is used if no such path exists.

The Minimum Drain Rate (MDR) scheme [8] introduced “the drain rate” metric. It is used together with the residual battery...
capacity in order to predict the lifetime of nodes based on the on-going traffic. In this the ratio of residual battery capacity and average consumption rate of a node represents how long the remaining energy can keep up the connections with current traffic condition. The maximum lifetime of a given path is determined by the minimum value of the corresponding cost function and the path with maximum lifetime is chosen. But this algorithm uses drain rate which is not easy to assess accurately and calculations can differ over small adjustments in assumptions. A number of other researchers addressed the energy utilization issues for routing in MANETs. Refs. [9–11] are examples of a few recent pieces of literature on this topic.

In our work, we considered the fundamentally important problem of energy-aware routing in ad-hoc networks. The efficient foraging behavior of naturally occurring small-sized and energy-constrained ants were studied in the past and it resulted in the theory of Ant Colony Optimization (ACO) [1]. ACO-based routing algorithms use control packets called ants, which collect information about a path, while travelling from source to destination node, and use this on its way back from the destination to the source. The ants also deposit pheromone to help future ants in the decision-making process. Each node contains a routing table. The routing table has for each destination node a list of entries, one for each of the neighbor node. These entries are a measure of the goodness of one neighbor over another one. These entries maintain the pheromone content using suitable variables. These variables are updated as ants traverse the various paths available in the network. As more and more path sampling ants are generated, there is a bundle of paths available with an estimate of their quality. The next hop is chosen stochastically, giving higher probability to links having higher levels of pheromone.

We observe that pheromone content can be use to choose the best paths out of a given network. It can also be used to stochastically route data. In this way, data for the same destination can be spread over multiple paths, with more data going towards higher quality paths, resulting in load balancing. The ants that traverse a network can be made explorative or localized by subtle variations in the way that they choose the next hop. All ACO-based algorithms employ these schemes to accomplish a particular task. The ant colony-based routing algorithm (ARA) [14], works mainly in an on-demand fashion, with ants setting up multiple paths between source and destination at the start of a data session. During the data session, data packets reinforce the paths they follow.

AntHocNet [13] is a hybrid multi-path algorithm designed along the principles of ACO-based routing in ad-hoc networks. We explain the main features of this algorithm below, the explanation of most of which are adopted with little changes from [13]. The algorithm has four major phases: reactive path setup, stochastic data routing, proactive path maintenance and exploration, and handling link failures. The reactive path setup phase has reactive forward ant-like agents launched by the source in order to find multiple paths to the destination. Similarly, backward ant-like agents return to the source to set up the paths. The path setup phase creates a number of good paths between the source and the destination, which can be used for subsequent routing. The forwarding of data packets takes place stochastically. When a node has multiple hops for the destination of the data, it will randomly select one of them with a probability calculated as a function of their quality. This procedure leads to automatic load balancing. The next phase is of proactive path maintenance and exploration where some control packets are sent in the network to improve the quality of the paths. In order to guide the path of the forward ants better, the algorithm also makes use of hello packets, which are short messages broadcasted by the nodes after a constant interval of time. These messages are also used to detect broken links. An updated view of its immediate neighbors are maintained by AntHocNet in order to detect link failures in the soonest possible time. It broadcasts a link failure notification message when a broken link is identified. Including the above, other ACO-based routing schemes such as AntNet [12] devised for ad-hoc networks, were not targeted towards energy conservation.

In this work, we propose an ACO-based multi-path routing protocol, named as Energy-Aware Ant-Based Routing (EAAR) protocol, which takes into account the various factors such as the power consumed in transmitting a packet and the residual battery capacity of a node to increase the battery life of the nodes by reducing the repetitive use of a selection of these nodes. We approach this problem by distributing the packets into multiple “good paths.”

We have considered the minimum battery energy remaining from the weakest node of the route and the hop-count of the route as our metric for path discovery. First, we included hop count in our consideration as the greater the number of hops, the greater will be the transmission, and the more will be the energy consumption. Second, the minimum battery remaining was taken into consideration after getting inspired from the success of MMBCR [6] in minimizing energy usage.

Although MMBCR is very effective in getting the energy-aware paths, it considers all the possible paths. We observed that this results in wasting too much of energy solely for route discovery. For this reason, we added the ACO-based approach in our scheme, which gets a number of good routes in a “better” way. Furthermore, the MMBCR approach transmits data on a single route, which puts all the data on only a few set of nodes. This results in more dead nodes, as only some nodes get used and the rest remain idle. However, our objective is to increase the battery life of nodes and thereby reduce dead nodes. So, we used multi-path transmission through the “good routes” only.

One of the advantages of multi-path routing is that, the data traffic is balanced and distributed while being transmitted from the source to the destination. Even if we have to transmit millions of data packet, multi-path routing ensures that there is less congestion as the whole load is distributed throughout the network, and better routes getting greater load.

2. Energy-aware ant-based routing

2.1. Principles of ACO

ACO uses the concept of artificial ants, analogous to the natural ants that behave as packets for ad-hoc networks. Ants are blind, so are the packets. Initially, they do not know where to head forward. So, they spread throughout the entire network with equal probability (in the same fashion as one would broadcast packets in a network).

A natural ant, while traveling in search of food, deposits a chemical substance called pheromone for other ants to sense and follow their path (such as packet forwarding through some metric in routing table at the network layer). While “combing” the area in all directions, if any ant reaches the food source (packet to destination), return ants are created, which head back towards the nest, reinforcing their traversed path, while going back to confirm that the path was successful. In this manner the discovery is done to find best possible paths, which have more ants traveled on it and as a result, more pheromone deposited on it.

Consider a network as in Fig. 1, where the green circle is the nest of ants and the red one is the food source. If we analyze, we will find that a typical route would contain a large number of cuts and turns. But for the sake of understanding we take that the ants can travel through yellow circles only.

Let us say that from the source node there are 3 paths. We distribute the population of foraging ants $P$ into 3 equal sections ($p_3$) such as: $p_1$, $p_2$, and $p_3$ going through the yellow circle connected.
to green one. This phenomenon is repeated at each yellow circle. On reaching the red circle, the ant returns back to the nest sensing the pheromone again. While going back, it deposits a stronger pheromone for other ants to follow its trail. This is the way the ants behave and we have used this behavior for the proposed energy-aware routing.

2.2. Motivation

The algorithm in this paper has been devised taking motivations from other ant colony optimization algorithms that perform better than other protocols. Algorithms like AnthocNet [13] and ARA [14], have performed better in many ways due to their proactive and iterative behavior. The iterative behavior makes the algorithm better than others like AODV, which do not consider frequently changing MANETs network. In a non-iterative protocol, the routes are found and broken very quickly as compared to iterative ant-based protocol. This makes the network more prone to errors. But the ACO algorithms reduce variability and error in network by choosing trusted path which have behaved well for quite some time.

Energy-aware protocols should have an algorithm so that they do not have to waste that much energy in finding new paths every time. So our paper tries to reduce energy consumption or wastage so to say, through discovering good paths right from the start. Also multi-path routing has its own advantages over single-path routing. They do not put more stress over some few chosen nodes. Instead they utilize the whole network. This is in a way helps in utilizing the energy of the whole network which is far superior to the single-path energy. Also most of the algorithms do not take into account the remaining power of a node while using it for transmission. This results in nodes getting off due to power shortage. Our paper also addresses this incorporating required procedure which we will discuss shortly.

2.3. The proposed scheme – energy-aware ant-based routing

2.3.1. Path discovery

The basic structure of the path discovery mechanism is inspired from AnthocNet [13], but, as will be evident from the following discussions, it is functionally distinct. Consequently, the explanation of some of the steps which are adopted from [13] are presented with little or no change. This is also true for other parts of the proposed scheme where there are resemblances with steps in AnthocNet. As in [13] when a source node, s, starts a communication session with a destination node, d, and it does not have routing information for d available, it broadcasts a reactive forward ant, say $F^i_d$, (here an ant is representative of a control packet). Due to the initial broadcasting, each neighbor of s receives a replica, $F^i_d$, which is $F^i_d$, k (the notation “.k” refers to indexing – the kth message of a single broadcast is represented as $F^i_d$, k).

After the next hop, the next neighboring node will receive $F^i_d$, k.l and so on where (k, l, . . . are integers). The task of each ant $F^i_d$, k.l.m.n . . . is to find a path connecting s and d. At each node, an ant is either unicast or broadcasted, depending on whether or not the current node has routing information for d. Since there will be no information initially, all are broadcasted at that point. Also, each packet maintains an array, J, in which its journey information is stored.

When a node receives several ants of the same generation, it will compare the path traversed by each ant to that of the previously received ants of the current generation. For example, let us suppose that an ant, A1, has a journey array, a set of nodes, J1, and another previously received ant, A2, had J2. If J2 is a subset of J1, then the packet is discarded immediately. If J1 is a subset of J2, the packet is accepted. In the third case, when both do not overlap completely, we use an acceptance factor, $\lambda$ (ranging from 1 to 2) if A2 had M hops and A1 currently has N hops, then we will allow only when $N < \lambda \times M$, where $\lambda$ is the factor by which each node takes a decision whether to accept or to reject a particular control packet that has more hops than the packet which came before it for the same destination. This means that if a packet has travelled more nodes than the packet arrived before, we will allow it only if it has not travelled more than twice the number in extreme case. This $\lambda$ factor can be changed according to requirement if one wants to reject more packets and reduce congestion in the network. But this decision is taken only if the packet is not the superset of any packet that is already received.

With this approach, we may get a set of paths with similar journey, but only some nodes will be distinct type of nodes. This will help us to get the “better” paths, because our parameter is not limited to hop count only. We attempt to find a path with the maximum of the minimum residual battery energy (MBR) of all the nodes of the journey – together we call it MMBR.

We take $\lambda = 1.5$, after experimenting with its different values. It is conjectured that an increase in the number of hops is likely to get us a “good” path in terms of MBR. We just cannot ignore hop count. So, we need to set the value of $\lambda$ by giving due consideration for both the parameters.

Using this scheme, overhead is limited by removing ants which follow “bad” paths, while there is still the possibility to find multiple “good” paths. We also obtain an increased number of paths, which implies that, if needed, in case of link failures, it will help us to divert the packet flow. Each forward ant keeps a list P of the nodes $\{1, \ldots, n\}$ it has visited.

On the arrival of the first reactive forward ant to the destination, we calculate the end-to-end delay for this first ant. The destination will wait for certain time $T_w$ for the rest of the reactive forward ants to come.

$$T_w = C_w \times T_d$$

In Eq. (1), $C_w$ is an integer factor and $T_d$ represents the end-to-end delay for the first received ant.

All the ants that came in this time are converted to backward ants as soon as they arrive and they travel back to the source retracing P (if this is not possible because of the absence of the next hop, for instance, due to node movements, the backward ant is discarded). While moving backwards to node i from node n and upon reaching there, each backward ant updates or makes an entry into the neighbors table of node i about $T^i_{n,d}$ which equals the inverse of the number of hops (H) multiplied by the minimum residual battery energy of all the nodes traveled up to the current node by the current backward ant (MBR):

$$T^i_{n,d} = \frac{MBR}{H}$$

In Eq. (2), $T^i_{n,d}$ is the value that a data packet will check when it arrives at node i, as to when it has to choose the next node.

We also drop the packets which have hop count of more than 10. The destination node will wait for a time Y times the end-to-end delay. If no data packet is received by the destination in this time, it will find the b next best paths, in addition to the previous one and...
then repeat the same procedure until a data packet is received in the given time.

### 2.4. Data session

Once the data session starts, the data packets are sent through the host. The host will either distribute the packet or the packet will choose the next node from the set of neighbors, \( N_i \), which have pheromone information in the table with probabilistic condition, \( P_{nd} \), as in AntHocNet [13]:

\[
P_{nd} = \frac{(T_{nd})^\beta}{\sum (T_{ld})^\beta}
\]

(3)

In Eq. (3), \( \beta \) is a factor which can take in a set of integer values. The traversal of each data packet increases the pheromone values of each link by a factor \( \pi \):

\[
T_{n,d} = T_{n,d} \times (1 + \pi)
\]

(4)

In Eq. (4), \( \pi \in (0,1/2) \). In the results reported in this paper, we took \( \pi = 1/10 \). The other nodes evaporate the pheromone deposits, resulting in a more frequent selection of better paths. Evaporation occurs periodically. For every \( \tau \) time period, node will evaporate the pheromone value, \( T_{n,d} \) automatically.

\[
T_{n,d} = T_{n,d} \times (1 - \rho)
\]

(5)

In Eq. (5), \( \rho \) lies between 0 and 0.5. As soon as the first proactive backward ant is received by the host, it again sends another forward proactive ant analogous to the first one. This eventually leads one to a better path.

#### 2.4.1. Route maintenance and link failures

While a data session runs, the routes are maintained through pheromone reinforcement and evaporation techniques. Our scheme does not waste energy by sending proactive ants.

When a link fails to transmit the packet and the node having it has no more neighboring nodes in the pheromone table to send, it sends a control packet to all the neighboring nodes, which have this node in their data to remove the current node’s entry. The node having the data then initiates a route request packet for the required destination and sends the data on the first arrival of its reply, without waiting. In the other cases of link failure, no action is taken.

### 2.4.2. Algorithm of EAAR

#### Algorithm EAAR

**Input:**

The following blank tables of all nodes are input:

1. The neighbor table: A table containing all nodes in the neighborhood of a node.
2. Seen table: A table containing all packets received by a node and their paths.
3. Routing table: A table containing next hop to transfer packets.

Initial pheromone for all nodes = 0.

**Output:**

Updated tables with all the values required to transmit data.

Pheromone value for selected nodes.

**Steps:**

1. Broadcast all the request packets and initialize a “seen” set \( S \) of every node as NULL.
2. On receiving any route request:
   - for all routes \( R_i \) in \( S \) of node check:
     - if the route traveled by the request is not a superset of the \( R_i \),
     - if the route is subset of \( R_i \) OR the hop count is less than 1.5 times the highest in the set \( S \):
       - add route in set \( S \) and rebroadcast it.
     - else
       - discard it
3. On reaching the destination, the route request is converted to the route reply, the path traveled is returned to, and the pheromone \( PH \) in the routing table of each node of path is added.

\[
PH = \frac{MBR}{HOPS}
\]

4. When the source receives the first reply, the delay of the first packet is made 5 times in order to receive more packets and the routing table is updated.
5. Data transmission is initiated with each packet, selecting next hop with probability \( P_{nd} \) from all available, by taking the pheromone values from routing table from Eq. (3).
6. On each transmission, the pheromone is reinforced and others are evaporated.
7. On link failure, Step 1 is repeated from the node that has data to send, but no neighbors available.

**Example:** The following explains the results of a dry run of the network shown in Fig. 2. In this section, we explain each step of the proposed routing scheme in detail.

---

**Fig. 2.** An example network topology.
In Fig. 2, intuitively, our most preferable paths should be the two shaded paths. One having minimum hop and the other having highest minimum battery, i.e., the path in which the node with the minimum battery, among others in the same path, is highest as compared to the minimum of the other paths. The red shaded route has all nodes having energy $\geq 5$ – we say that the minimum battery of that route is 5. Similarly, for other routes, every node has energy greater than or equal to some positive number, which results to be less than 5. So, the route with the highest minimum battery (MMBR) is the red colored one having MMBR as 5 units and the route with the minimum hops is the purple colored one. The overall functioning of the algorithm step by step is explained below.

In Fig. 3, we show how the network performs. For the given scenario, we broadcast the $\mathcal{F}_{d}^k$ control packet (where $k \in \{1,2,3,4,5\}$). We give them numbers according to their transmitting timestamp at node 6. In this example, packets are given names $\mathcal{F}_{6}^{17}.1$, $\mathcal{F}_{6}^{17}.2$, $\mathcal{F}_{6}^{17}.3$, $\mathcal{F}_{6}^{17}.4$, $\mathcal{F}_{6}^{17}.5$ for nodes 10, 1, 2, 3, and 5, respectively. Each node, upon receiving this control packet, checks its cache to see if any other control packet of the same broadcast has been received earlier or not. If not, then it will accept the packet and add it to its cache. Else, it will discard the packet.

This functioning of packet discarding is explained in detail below. Now each control packet, which has been received at different nodes, generates another set of packets for further discovery, as it has not reached the destination. Each of these broadcasts will

<table>
<thead>
<tr>
<th>Node</th>
<th>Battery</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td>14</td>
<td>9</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>6</td>
</tr>
<tr>
<td>20</td>
<td>7</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
</tr>
<tr>
<td>22</td>
<td>8</td>
</tr>
<tr>
<td>23</td>
<td>8</td>
</tr>
<tr>
<td>24</td>
<td>7</td>
</tr>
<tr>
<td>25</td>
<td>7</td>
</tr>
</tbody>
</table>

In Fig. 4, we show how EAAR works.
produce packets named as say, X,Y, where X is the name of the control packet received by the node and Y is the index associated with all new control packets. For example, the control packet at node 3 (named as 4) will produce control packets \(F_{6}^{17}, F_{6}^{17}, F_{6}^{17}, F_{6}^{17}, F_{6}^{17}, F_{6}^{17}, F_{6}^{17} \), and so on.

Let us now discuss how a node discards or accepts a packet. For example, in Fig. 4, the node 4 gets two packets one after another (order does not really matter in this case). It will accept both \(F_{6}^{17}, F_{6}^{17} \) and \(F_{6}^{17} \). As it does not have any packet from the broadcast of 3 and 5, it will accept both of them, as the route traveled by both is different, and not subset of any one of them. This is the basis of the criterion, which recommends that compare the nodes traveled for any two packets and see if they are equal or subset of one another.

In Fig. 5, we see that when node 3 gets the packets named \(F_{6}^{17}, F_{6}^{17} \) from node 2, it will discard it. This is because a packet from the original broadcast (packet named 4) from node 6 has reached it earlier. So, all the packets, like \(F_{6}^{17}, F_{6}^{17} \), would be rejected. However, if 3 is reached before packet \(F_{6}^{17} \), it will be accepted and \(F_{6}^{17} \) would too be accepted. In the process we are saving two paths one shorter, and the other having smaller transmission time. Although our intention was for the former and memory based paths, but if a smaller transmission path comes in the process, it will benefit the overall network fault tolerance.

In the other case too, at node 11, packets \(F_{6}^{17}, F_{6}^{17} \) and packet \(F_{6}^{17} \) are received. \(F_{6}^{17}, F_{6}^{17} \) is the product of the control packet at node 7 (named \(F_{6}^{17}, F_{6}^{17} \)), which is the product of the packet from node 10 (packet named \(F_{6}^{17}, F_{6}^{17} \)), which was first produced by the source node 6. Similarly \(F_{6}^{17}, F_{6}^{17} \) is produced by the nodes 6, 10, and 11, respectively in the chronological order. Now node 11 would reject \(F_{6}^{17} \) if it has received \(F_{6}^{17}, F_{6}^{17}\) whose path is the subset of \(F_{6}^{17}, F_{6}^{17}\)’s. In this way we can control the unwanted control packets from the network and help in reducing the traffic.

Finally, on reaching the destination, let us say, x, y, z are the names of packets based on which the paths are considered and then compared. The return ants/packets are then generated going in the opposite direction and depositing pheromone hop by hop, on all the intermediate nodes. As a result, it enters a numerical pheromone \(T_{n,17}^{17} \) value in the routing table of each node i according to the criteria as explained above. For example, for the return packet, w, which travels from Node 22 to Node 15, having MBR 5 (considering Nodes 17 and 22) and the hops(H) = 1 (from 17 to 22), we have:

\[ T_{22,17}^{15} \]

There are many more return ants created and some of them carry pheromone to the path listed below. As we can see, many paths are discovered, but if we look closely, it is just the next hop which matters the most, and in all cases it is around 2 or 3. The bold ones have the maximum pheromone value. These paths correspond to the colored paths in Fig. 3, which by the way are the paths on which the data transmission should take place. So our algorithm fulfills our objective which is to impart maximum pheromone value to these paths so that they can be used for data transmission more than any other path.

### Simulation results and discussion

In this section, we report the simulation experiments we conducted and the results we obtained from comparing the perfor-
Fig. 6. On reaching the destination node.

performance of EAAR with a selection of the major benchmark algorithms. Simulation was done using GloMoSim tool [15,16]. We used the PARSEC [17] compiler as a C [18,19]-based simulation language.

The following parameters [20] were set for the simulations performed:

(a) SIMULATION-TIME: 1000 s. This parameter represents the maximum time given for the whole simulation to last. All the results are obtained within this allotted time.

(b) TERRAIN-DIMENSIONS: (2000 m × 2000 m). It is the area in which all the simulation takes place. All the nodes and their movement are confined to this area. We have taken it to be 2000 m in length and breadth.

(c) NUMBER-OF-NODES: 30. This parameter represents the number of nodes that take part in the simulation. We have taken this value to be thirty. We have considered a medium sized network consisting of 30 nodes. The density of the network may vary with time due to the mobility of the nodes.

(d) NODE-PLACEMENT: UNIFORM. This represents the node placement strategy. We have taken it to be according to the uniform distribution. Initially, the nodes will be uniformly distributed and would change their positions randomly according to the mobility mode.

(e) PROPAGATION-LIMIT: −111.0 dBm. This is the propagation limit beyond which the signals will be accepted. Signals below this parameter (in dBm) are not delivered. This value must be smaller than RADIO-RECEIVING-SENSITIVITY + RADIO-ANTENNA-GAIN of any node in the model. Otherwise, the simulation results may be incorrect. Lower value should make the simulation more precise, but it also makes the execution time longer.

(f) PROPAGATION-PATHLOSS: TWO-RAY. The PROPAGATION-PATHLOSS parameter specifies the path loss model. The Two-Ray Model [21,22] considers both the direct path and a ground reflected propagation path between the transmitter and the receiver. This model predicts that the received power falls off with distance raised to the fourth power, or at a rate of 40 dB/decade.

(g) RADIO-TYPE: RADIO-ACCCNOISE. This is the radio model to transmit and receive packets. RADIO-ACCCNOISE refers to the standard radio model.

(h) RADIO-FREQUENCY: 2.4e9. This represents the frequency of radio transmission used, which for our simulation has been taken to be equal to 2.4e9 Hz.

(i) RADIO-BANDWIDTH: 2,000,000. Bandwidth = 2,000,000 bits per second.

(j) RADIO-RX-TYPE: SNR-BOUNDED. This parameter specifies the packet reception model. In the SNR-BOUNDED mode, if the Signal-to-Noise Ratio (SNR) [21,22] is more than the RADIO-RX-SNR-THRESHOLD (in dB), it receives the signal without error. Otherwise the packet is dropped.

(k) RADIO-RX-SNR-THRESHOLD: 10.0. This is the threshold SNR for reference. We have used it as 10.0.

(l) RADIO-ANTENNA-GAIN: 0.0. This is the Antenna Gain (in dB). The Antenna gain is a measure of the directionality of an antenna. The Antenna gain is defined as the power output, in a particular direction, compared to that produced in any direction by a perfect omni-directional antenna.

(m) RADIO-RX-SENSITIVITY: −91.0 dBm. This is the sensitivity of the radio (in dBm). The sensitivity is the measure of the weakest signal that may be reliably heard on the channel by the receiver (it is able to read the bits from the antenna with a low error probability). This indicates the performance of the receiver, and the lower the value the better the hardware.

(n) MAC-PROTOCOL: 802.11. This signifies the MAC protocol to be used for the simulation. We have used IEEE 802.11 [23,24] protocol for the MAC layer.

(o) INITIAL ENERGY: ALL NODES EQUAL. Having equal energy for all nodes gives us the scope to experiment more within a given network. Every path and every node gets equal opportunity to compete for energy.

(p) DATA SIZE: VARIABLE. We have not defined a particular data packet size. We have used the following two different sizes:

(i) Small (100 times the control packet).

(ii) Large (125 times the control packet bytes).

(q) MOBILITY: VARIABLE. Mobility is changed depending upon scenarios.
The parameters specific to Ant Colony Optimization were also set. Ant Colony Optimization (ACO) is a paradigm for designing metaheuristic algorithms for combinatorial optimization problems. The first algorithm which can be classified within this framework was presented in 1991 [26,27]. This algorithm is used in EAAR and ANTHOCNET as the base algorithm. The following parameters are set which correspond to Eqs. (3)–(5) for EAAR as described in the earlier sections of this paper.

\[ \beta = 1, \quad \pi = \frac{\tau_{i,j}}{10}, \quad \rho = 0.1. \]

The various traffic types for the Constant Bit Rate (CBR) used are as follows:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Packets dropped per packets delivered</th>
<th>Number of dead nodes</th>
<th>Total energy consumed</th>
<th>Energy per packet delivered</th>
<th>Packets delivered per dead node</th>
<th>Packets dropped per packets delivered</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following benchmark protocols were used for comparing the performance of our proposed protocol, EAAR: AODV [2], MMBCR [6], and AntHocNet [13]. AODV is one of the most popular and widely used proactive routing protocols for routing in ad-hoc networks. MMBCR is very popular for energy-aware routing and AntHocNet is perhaps the most popular ACO-based ad-hoc network routing algorithms. This justifies the choice of these benchmark algorithms.

The simulation experiments were done under the following six scenarios. In the context of the Random Waypoint Mobility Model [25], WP-PAUSE refers to the amount of time the nodes will pause and WP-MIN-SPEED and WP-MAX-SPEED refer to the minimum and the maximum speeds, respectively, with which the nodes will move.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Traffic Type</th>
<th>Parameters</th>
<th>Number of packets dropped</th>
<th>Number of Dead Nodes</th>
<th>Total Energy Consumed</th>
<th>Energy Per Packet Delivered</th>
<th>Packets Delivered Per Dead Node</th>
<th>Packets Dropped Per Packets Delivered</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CBR</td>
<td>1</td>
<td>26</td>
<td>100</td>
<td>1536</td>
<td>0S</td>
<td>5</td>
<td>0S</td>
</tr>
<tr>
<td>2</td>
<td>CBR</td>
<td>12</td>
<td>18</td>
<td>100</td>
<td>1536</td>
<td>15</td>
<td>250S</td>
<td>0S</td>
</tr>
<tr>
<td>3</td>
<td>CBR</td>
<td>23</td>
<td>9</td>
<td>100</td>
<td>1536</td>
<td>15</td>
<td>500S</td>
<td>0S</td>
</tr>
<tr>
<td>4</td>
<td>CBR</td>
<td>14</td>
<td>27</td>
<td>100</td>
<td>1536</td>
<td>15</td>
<td>750S</td>
<td>0S</td>
</tr>
</tbody>
</table>

Several simulation experiments were performed. Some of the key results obtained are reported below.

Fig. 7 shows the relative performance of the four protocols with respect to the number of packets dropped. As we can see, EAAR has very less number of packets lost compared to the others. This observation is very prominent for the 3rd, 4th and the 5th scenarios. Let us take these scenarios one at a time.

In the first scenario, when the mobility is zero and the packet size is small, the network acts as an ideal one. As a result, every protocol behaves similarly. But in practical terms, this is not possible all the time. In the second scenario too the difference is very minute. But in the third scenario, when we introduce some mobility, the difference is noticeable. The packet drop for the ant-based protocols (AntHocNet and EAAR) is found to be minimum. This is due to multi-path routing. But even then EAAR performs better than AntHocNet, which is designed for the better delivery of packets. In the fourth scenario, for large data sizes, the difference increases and EAAR is a clear favorite. This shows that the performance of EAAR improves for bigger data sizes. For instance, for the 4th scenario, the number of packets dropped in the case of EAAR is around 5, whereas the number of packets dropped for AODV, AntHocNet and MMBCR are approximately 70, 110 and 90, respectively. This is due to the fact that it has a good route maintenance system as well as a better link repair mechanism and lesser number of dead nodes, which leads to reliable transfer of packets. However, the other protocols lose relatively larger number of packets due to the absence of good repair system in the presence of dead nodes. The 5th and the 6th scenarios just show the consistency of EAAR in any scenario.

Fig. 8 shows that EAAR, in general, has a relatively very less number of dead nodes compared to the benchmark protocols. This is perhaps related to the existence of multi-path routing in EAAR. For
the same reason, AntHocNet also shows less number of dead nodes, but it cannot beat EAAR because EAAR is energy-aware. As AODV and MMBCR route using the unipath mechanism, they lose more nodes because of the faster depletion of energy from these selected nodes. Let us discuss each scenario.

In the first scenario, MMBCR performs best among all the protocols considered, followed by EAAR. In this scenario, the packet size is small with no mobility. This gives MMBCR an edge, as it does not consider mobility much in its approach. Due to smaller data sizes, the energy lost is not substantial. This is the reason MMBCR shows to perform relatively well. In the second scenario, we can see the consistency in the ant-based routing protocols (EAAR and AntHocNet) as they are multi-path and offer better performance. But still EAAR performs better. In the third scenario, when we introduce mobility, we can see that there is still less number of dead nodes in EAAR as compared to other protocols. This shows how well EAAR adapts itself in the mobile environment. In the fourth scenario too, we see a similar result. So is the case in the 5th and the 6th scenario as well. Thus, we can infer that in every mobile scenario, the performance of EAAR is best as compared to the other mentioned protocols.

AODV was devised to get the best throughput. Consequently, it has a greater number of packets delivered compared to the other protocols. This is shown in Fig. 9. As our scheme waits for some time to get more routes and then transmit, the packets delivered are relatively less compared to AODV, but it is still better than AntHocNet, in which the mobile network loses more links, thereby impairing the ability to deliver packets. Let us discuss the variation in each scenario.

In the first scenario with no mobility, AODV and MMBCR perform better than the multi-path routing protocols. So is the case in the second scenario too. This is due to the fact that in a static environment, once a connection is established, it hardly breaks. So, AODV and MMBCR perform well in these scenarios. But as we introduce mobility in the environment, the performance of these protocols decreases substantially, whereas EAAR performs as good as previously. In scenarios 1, 3 and 5, where the data size is small, we see a slight variation for EAAR. So is the case for the scenarios 2, 4 and 6. But the other protocol’s performance degrades as we increase mobility. AODV being the exception, for the reason mentioned above, it delivers more packets throughout all scenarios.

As we can see from Fig. 10, the energy consumed in the overall network is minimal in EAAR, although the packets delivered remain very similar. This affirms that, in all conditions, EAAR is the most energy-aware compared to the existing protocols.

In scenarios 1 and 2, all the protocols perform equally as there is no mobility and hardly any connection breaks. But still multi-path routing schemes perform better. But in a mobile environment, where nodes move randomly, multi-path routing outshines single routing algorithms. Because of their multiple paths, if a route breaks, another route is used simultaneously to transport the load of the broken route. Further, as we introduce more and more mobility, in scenarios 5 and 6, the packet delivered by AntHocNet decreases and so is the energy consumed. If one can notice the consistency of EAAR in every environment, one has to come to the conclusion, that EAAR is the most energy-aware among all the protocols mentioned. One more point to be noted here is that in every even scenario (such as 2, 4, and 6) the energy consumed is slightly more because of the larger data packet size.

As we can see, not only EAAR, but also AODV has this ratio less only in most mobile conditions, because of the energy saved in the control packets. EAAR uses more packets to find routes and in a more mobile state more route discovery is needed.

In scenarios 1 and 2, we observe that all the protocols perform equally well, because there is no mobility to cause problems. Nevertheless, as we add mobility to the environment, we find AODV and EAAR perform better than the other two protocols. Also EAAR performs better in scenarios 3 and 4. In scenarios 5 and 6, as we increase mobility, due to fast delivery of packets by AODV, this ratio decreases for it, while it increases for EAAR. But as we have seen in the previous results the number of dead nodes increases for AODV, therefore it sacrifices more energy for less packets.

In Fig. 12, we have shown how much each scheme sacrifices nodes for packets delivered. As we can see the higher the ratio...
In the first scenario, MMBCR, being single-path energy-aware protocol, performs the best. Due to static environment and small data packet size, it does not have any problem in transferring the packets. But as we increase the packet size, we can see a decrease in its performance. As we increase the mobility of the network in the scenarios 3 and 4, EAAR is a clear leader followed by AODV. EAAR, unlike other protocols, delivers a relatively constant ratio as compared to the other protocols whose variations are much higher.

In Fig. 13, we show how each protocol successfully delivers a packet with respect to the packets dropped. In other words, it gives an idea about the throughput of the protocols. The lower the ratio, the better it will be.

In scenarios 1 and 2, we see all the protocols to be equally good, but as we introduce mobility, the value changes. While we can still see a consistent EAAR performance throughout, we see sharp rise in the ratio of MMBCR and AntHocNet, while AODV still performs better than these two, but not as good as EAAR.

In all cases, the performance of EAAR is best while AntHocNet, being an ant-based multi-path scheme, also performs well in some cases. The reason for this is attributed to the multi-path routing schemes. As there are more than one path to transmit, we see lower packet drops with respect to the packets delivered.

4. Concluding remarks and future scope

It is evident from all the observations that EAAR is better compared to AODV, AntHocNet and MMBCR in most of the mobility and energy conservation scenarios. The superior energy conservation attribute of EAAR, in turn, helps in extending the life of the nodes, which further helps in increasing the lifetime of the network.

The energy consumed in the network, the energy per packet and the packets lost in the case of EAAR are less compared to others in small and medium mobility scenarios. The packet delivery ratio is, therefore, much superior for EAAR. Of course, the packet delivery rate and the energy per packet in high mobile conditions are relatively not as superior for EAAR as the other parameters. This is because energy-awareness increases the time to judge the best route for transmission. The results get better for larger data packets.

From the simulation results one can surely say that EAAR is very energy-efficient because of its following features:

- Multi-path routing.
- Ant-based path discovery for energy abundant routes.
- Good route maintenance and recovery mechanism.

In the future, we plan to:

- Use other mobility models, other than random-waypoint.
- Use networks having a very large number of nodes and varying the network density. In this paper, we have considered the network to be comprised of less than 100 nodes.
- Develop test-beds for observing whether the results remain the same when they are applied to practical scenarios using various wireless devices and creating an ad-hoc network.

Acknowledgements

Part of this work done by the first author was partially supported by a grant from the Council of Scientific and Industrial Research (CSIR), India, Grant Ref. No. 22(0477)/09/EMR-II, which the author gratefully acknowledges.

References


Sudip Misra is an Assistant Professor in the School of Information Technology at the Indian Institute of Technology Kharagpur, India. He received his Ph.D. degree in Computer Science from Cornell University (USA), Yale University (USA), Nortel Networks (Canada) and the Government of Ontario (Canada). He has served as a reviewer for international journals as well as a keynote speaker at conferences. His current research interests include the design and product engineering roles. His current research interests include wireless ad-hoc networks.

Pushkar Gupta did his bachelors in Information technology from Netaji Subhas Institute of Technology, Delhi University, New Delhi, India. He is currently doing his masters in Management from ESCP Europe and has completed his masters from Management Development Institute, Gurgaon, India. He has done research in the field of mobile ad-hoc networks and has been co-author for 3 scholarly research papers at international conferences. His current interest include research and analysis of IT sector for global financial and consulting firms.

Karan Verma received his B.E. degree in Information Technology from Netaji Subhas Institute of Technology, Delhi University, New Delhi, India. Currently he is working as a Development Engineer with the IOS-XR team of Cisco Systems at its Bangalore campus in India. At Cisco he has been working with the development and maintenance of the Embedded Event Manager which is a manageable component of IOS-XR. His current research interests include design and analysis of network protocols, mobile ad-hoc networks, architecture & design of NG routers & switches. He has researched in the field of mobile ad-hoc networks and has been the co-author for 3 scholarly research papers at international conferences and also served as a reviewer for international journals.
Prayag Narula did his Bachelor in Engineering from Netaji Subhas Institute of Technology, University of Delhi in the field of Information Technology. He is currently a graduate student in the School of Information at the University of California, Berkeley, USA. He was a researcher at Helsinki Institute for Information Technology (HIIT) as a member of the Ubiquitous Interactions research group, where he worked in various EU funded projects. His areas of interests are Human Computer Interactions, Multimodal Interaction techniques, Mobile Computing and Mobile Ad-hoc Networks.