Design of an Energy-Efficient and Reliable Data Delivery Mechanism for Mobile Ad Hoc Networks: A Cross Layer Approach

Mohammad Mehedi Hassan1*, S. M. Kamruzzaman1, Atif Alamri1, Ahmad Almogren1, Abdulhameed Al-Elaiwi1, Md. Manowarul Islam2 and Md. Abdur Razzaque2

1 College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
2 ICT Department, Mawlana Bhashani Science and Technology University, Bangladesh
3 Computer Science and Engineering Department, University of Dhaka, Bangladesh

SUMMARY

In a mobile ad hoc network, the data packet may fail to be delivered for various reasons mostly for route failure, congestion and battery energy drain. Hence, providing reliable and timely data delivery in this network in an energy-efficient way is challenging. Although there exist several solutions to solve these problems, they can handle either route failure or congestion or energy-efficient routing. Hence, to cope up with all the problems simultaneously, we propose a route failure and congestion-aware energy-efficient cross layer design (RCECD) that spans the transport and network layer. In the transport layer, we introduce the concept of local packet buffering during link failure and congestion. As a result, packet dropping rate of the network as well as energy consumption decreases. In the network layer, a routing protocol is proposed for selecting the energy-efficient path for data transmission. It uses the buffering mechanism in case of route maintenance. In addition, we employ a multi-level congestion detection and control mechanism at the source and intermediate nodes that can judiciously take the most appropriate decision for congestion control in the network proactively. The simulation results showed that the proposed cross-layer design provided better performance as compared to the state-of-the art protocols.

KEY WORDS: Mobile Ad hoc network, cross layer design, reliable data delivery, route failure, congestion control, and energy-efficient routing

1. INTRODUCTION

A mobile ad hoc network (MANET) [1] is a non-infrastructure network of mobile nodes connected by a wireless link. The nodes are normally battery-powered, hence, have small transmission ranges and use multi-hop transmission routes for long distance data transfer. Due to dynamic mobility of the nodes, time-varying wireless channels and battery energy exhaustion in mobile nodes, the MANET routes are often unstable, causing routes to fail frequently and reduce the network lifetime. Another important problem in MANETs is the dropping of data packets due to congestion in the network. Congestion may occur due to failure of link, queue overflow or channel or media overloading [2]. The congestion leads to packet losses, throughput degradation of networks, and wastage of time and energy for congestion recovery. In addition, since there is no fixed infrastructure, all nodes in a MANET share a single transmission channel; many nodes may contend for the channel
simultaneously to transmit data packets, increasing packet collisions in the network. In such a situation, the congestion collapse [3] may occur when no node will be able to transmit their data packets. Therefore, achieving reliable, energy-efficient, and timely data delivery is a challenging problem in mobile ad hoc network.

Several routing protocols [3][4][5][6][7][8][9][10][11] have been suggested in recent years to handle route failure as well as congestion and support reliable data transmission in an energy-efficient way. Some of these protocols [4][5][12] use backup route(s) on the failure of primary route. However, the maintenance of multiple alternative paths is difficult, costly, and time-consuming. Other routing protocols [3][6][7][8] uses multiple routes to balance traffic loads on the event of congestion or route failures and thus improve the network performance. However, the problem is that multiple route maintenance overhead affects the network performance significantly. Several other research works [9][10] focus on handling link failure of MANETs using local recovery process. However, packets might be dropped at the intermediate nodes if the local route discovery takes longer period of time. There are some researches [11][13][14] on using packet buffering or route caching to handle link failure. However, these approaches require a separate memory module which is not only costlier but also not implementable for all devices in the network.

Some other works [15][16][17][18][19][20] focus on designing energy-efficient routing protocols for MANETs. Some of them consider the remaining energy of nodes and the energy consumption of data transmissions in designing of the routing protocols, whereas, others consider balancing traffic load and the proportion of successful data transmissions [17][18][21]. However, the design of energy-efficient routing protocol to reduce energy consumption and prolong network lifetime requires consideration of other important factors, such as avoiding critical nodes having low residual energy, ensuring end-to-end data reliability from intermediate nodes other than source, and limiting broadcast messages throughout the network. Recently, cross layer-based routing protocols are being suggested for MANETs [21][22][23][24]. Some of them address the link failure and energy-efficient routing problem of MANETs by combining MAC and network layer [21][23]. Others consider to combine Physical, MAC and network layer such as [22]. To address congestion, MAC, network and transport layers were combined in [24].

However, all the aforementioned approaches can handle either congestion or route failure or energy-efficient routing. To cope up with these problems, in our previous paper [13][25], we proposed a route failure and congestion-aware data delivery mechanism for the MANETs. In the transport layer, we introduced the concept of local packet buffering when needed. The nodes on an active route buffer incoming packets at their local transport layer queues (TQs) during link failure and, on finding a new path, resume their transmissions. In addition, we used a multilevel congestion detection and control mechanism at the source and intermediate nodes for congestion control in the network proactively. However, the previous paper did not consider the energy efficient reliable data delivery which is very important for energy-constraint MANETs. Therefore, in this paper, we extend our previous work by proposing a routing protocol in the network layer for selecting the energy-efficient path for data transmission, based on minimal residual path energy and hop count. Thus, our approach can handle both congestion and link failures in an energy-efficient way, and ensures end-to-end reliable and efficient data delivery. The contributions of this work are summarized below:

- We propose a cross-layer buffering capability of nodes, spanning through network and transport layers when link failure and congestion occurs.
- In the network layer, an energy-efficient routing protocol is presented that can utilize the TQ buffering mechanism for route maintenance and local path repair.
- The proposed energy-efficient scheme can balance the energy consumption throughout the network. In addition, it reduces energy consumption by limiting broadcast messages. Further, it avoids critical nodes, having low residual energy, to participate in routing and can select the best route from a number of routing path.
- An efficient congestion control mechanism spanning network and transport layer is proposed, where nodes can detect multiple congestion levels of the network and take proper control actions by using the buffering mechanism of transport layer to reduce the packet dropping.
Various simulations were carried out based on different traffic loads and route failure rates using NS2 simulator to evaluate the performance of the proposed approach. The results demonstrate that our proposed design outperforms a number of state-of-the-art approaches in terms of packet delivery ratio, average end-to-end packet delay, energy-efficiency, throughput, and normalized routing overhead.

The rest of this paper is organized as follows. We describe related works and motivation in Section 2 and Network model and assumptions in Section 3. Our proposed cross layer design for reliable data delivery mechanism, RCECD, is presented in Section 4 and the performance evaluation are presented in Section 5. Finally, we conclude the paper in Section 6.

2. RELATED WORKS AND MOTIVATION

A significant research effort has been observed in recent years on handling route failures, congestion and energy-efficient routing in mobile ad hoc networks. We can categorize them into six different types based on their working principles.

The first category of works use backup routes on the failure of primary route. In AODV with Backup Routing (AODV-BR) [4] and Multi-path AODV, AOMDV [12], source nodes create alternative routes to the destination and on failure of any one of them nodes deliver data packets using an alternative route; however, they suffer from two problems: stale route and duplicate packet transmission. In AODV-Based Backup Routing Scheme, AODV-BBS [5], each node maintains two hop neighborhood information for finding alternative routes. But, the maintainace of multiple alternative paths is difficult, costly and time-consuming, which turn reduces the network efficiency.

The second category of routing protocols use multiple routes to balance traffic loads on the event of congestion [3] or route failures and thus improve the network performance. For example, a distributed Multi-path DSR protocol (MP-DSR) [6] improves QoS with respect to end-to-end reliability; Split Multi-path Routing (SMR) [7] uses multiple routes to split traffic and mitigate congestion; nodes in congestion adaptive Routing Protocol (CRP) [8] use bypass routes to mitigate congestion, etc. But, the problem is that multiple route maintenance overhead affects the network throughput.

The third category of works focus on link failure of MANETs using local recovery process. Most of them try to find out a local path whenever a link failure occurs in the network. For example, in Local Reapair AODV based on Link Prediction, LRAODV_LP [9], if a node detects that the signal strength goes below a predefined threshold, it initiates a fresh route discovery rather than sending error message backward. In Implicit Back-up Routing-AODV (IBR-AODV) [10], a neighbour of an active route temporarily stores (overheard) data packets and acts like a back-up node. Whenever any link failure occurs in the network, this back-up node creates new route to the destination and sends data packets. Here, the problem is that many neighbour nodes have to store data packets unnecessarily. Also more than one neighbour nodes may try to send the same data packet. This results in duplicate packet transmission of data packets and reduces network efficiency.

The fourth category of works handle link failure using packet buffering or route caching. In [11], a two hop routing (THR) protocol is proposed, in which if a node needs to transmit data it first checks its own routing table which contains route of two hop distance nodes. If any route is found only then data packets are delivered; otherwise, a fresh route discovery process is initiated. If route failure occurs, the intermediate nodes starts packet buffering in a separate physical memory module. However, the requirement of a separate memory module is not only costlier but also not implementable for all devices in the network.

The fifth category of works focus on designing energy-efficient routing protocols for MANETs [15][16][17][18][20]. For example, [15] and [16] consider the remaining energy of nodes and the energy consumption of data transmissions in designing of the routing protocols, whereas, others consider balancing traffic load and the proportion of successful data transmissions [17][18][21]. However, the design of energy-efficient routing protocol to reduce energy consumption and prolong network lifetime requires consideration of other important factors, such as avoiding critical nodes

having low residual energy, ensuring end-to-end data reliability from intermediate nodes other than source, and limiting broadcast messages throughout the network.

The sixth or more recent category of works focus on cross layer-based routing protocols for MANETs. For example, to address the link failure problem in MANETS, a cross Layer route optimization using MAC overhearing is proposed in [23]. Kumar et al. [22] proposed a three cross layer scheme (physical, MAC and network) to reduce the link break in MANETs. For congestion control, Reddy et al. [24] proposed a cross layer approach where the MAC and network layers have dynamic functionalists in identifying the congestion and standardization. The function of the transport layer is distinguished in bearing the congestion i.e. congestion endurance. To address energy-efficient routing, a cross-layer design is proposed in [21]. For the network layer, a minimum transmission energy consumption routing protocol was developed. For the MAC layer, an adaptive contention window (ACW) was used to dynamically adjust the back-off time.

Unlike the aforementioned works, we address all the problems of route failures, congestion and energy-drain in the network simultaneously in this work, and our approach has some unique features, such as, buffering data packets at transport layer queue, balancing the energy consumption throughout the network and detecting as well as controlling multi-level congestion in the network. Thus, our approach can ensure end-to-end reliable and efficient data delivery in the MANETs.

3. NETWORK MODEL AND ASSUMPTIONS

We consider an energy-constraint mobile ad hoc network (MANET) having a large number of nodes, where each node moves randomly and can act both as a router and a host simultaneously. We consider the IEEE 802.11 DCF as the MAC layer protocol. Each node in the network broadcasts HELLO massage periodically to its one hop neighbors. Each node that receives HELLO message updates their neighbor table. Besides, since nodes in MANETs are self-organizing and independent of each other, they can move randomly and frequently. As a result, the link failure may be occurred. Now, there are many ways to detect link failure, for example, using periodic HELLO messages [26] or link layer feedback [27], [28]. These HELLO messages are local advertisements for the continued presence of the link.

As mentioned earlier, the nodes in our proposed mechanism exchange HELLO messages periodically to ensure link connectivity. The following two parameters are associated with a HELLO message: HELLO INTERVAL is the maximum time interval between two consecutive HELLO message transmissions and HELLO LOSS allowed is the maximum number of loss of HELLO messages that a node can tolerate before it declare the link breakage. If a node does not receive any HELLO message from its neighbor node within HELLO LOSS allowed × HELLO INTERVAL, then the node assumes that the link is not available for data transmission.

4. THE PROPOSED CROSS LAYER APPROACH

In this paper, we propose a route failure and congestion-aware energy-efficient cross-layer design for reliable data delivery in the MANETs. In particular, a cross layer scheme is developed by combining the transport layer and the network layer. In the transport layer, we introduce the concept of local packet buffering during link failure and congestion. In the network layer, an energy-efficient routing protocol is proposed that utilizes the buffering mechanism in transport layer for route maintenance. Besides, an efficient congestion detection and control mechanism is proposed that spans transport and network layer, where nodes can detect multiple congestion levels of the network and take proper control actions to reduce the packet dropping. For a heavily loaded network, nodes will send an ALERT message to previous nodes not to increase the data forwarding rate. In a heavily congested network, nodes buffer their incoming packets to reduce packet loss rate. Thus, the proposed cross layer approach can control the congestion in MANET proactively. In the next sections, we describe the proposed cross layer scheme in details.
4.1. Packet Buffering Using Transport Layer Queue

In MANETs, generally a mobile node can act as both as a router and as a host. Hence, in our cross layer design, all intermediate nodes use a separate transport layer queue (TQ) to store incoming data packets, when needed. We assume, as long as there is no link failure or the network does not become heavily congested, nodes act like a conventional router and simply forward data packets. But when the network is heavily congested or link failure occurs, nodes use their TQs for buffering incoming packets.

Figure 1 shows the node A is acting as router in normal operation (indicated by solid line). After receiving packets, it just forwards the packets to next hop using lower three (physical, data link and network) layers of TCP/IP model. But in case of link failure, node A uses its transport layer queue to buffer data packets (indicated by dotted line). If a new partial path is found, node A starts its transmission process from transport layer queue as well as normal delivery of packets. The dotted lines indicate performing a transmission process from local TQ and solid lines indicate normal data delivery process.

Now a question arises-why does our approach use separate queue at transport layer for buffering the packets? Many source nodes in a MANET might deliver data packets to many other destination nodes simultaneously. An intermediate node may work as a forwarder of many such sourcedestination pairs, and it needs to store incoming data packets in the network layer queue for a very small period of time before forwarding the packets from the queue by examining the addresses of corresponding destination nodes. However, in case of link failure or congested state, nodes have to store data packets for relatively longer period of time since they have to wait for a partial path. For this reason, storing data packets into network layer queue during link failure or congestive states might hamper packet forwarding of other good connections and thus cause queue overflow at the node. As a result, packet dropping rate at the node will increase and overall throughput of the network will decrease. Furthermore, storing data packets at the transport layer during link failure may facilitate the intermediate node to ensure the end-to-end reliability from that node, rather than from the source. In this case, the intermediate node does not stamp any new sequence number with the stored packets and takes special care for their delivery. For the aforementioned reasons, an intermediate node uses separate queue in transport layer to buffer incoming data packets, allowing smooth packet forwarding of other connections through the node.

We define a cross-layer interface between network layer and transport layer as shown in 2. The interface has two components: receive interface R and delivery interface D. The interface R receives the packets from network layer queue and puts them into the transport layer queue when a link failure or congestion occurs. Similarly, all the intermediate nodes buffer the packets in their transport layer queues for that corresponding destination using their cross-layer interfaces. Whenever, a partial path is found, the network layer of the node informs transport layer through the interface. Then the interface D delivers the data packets to network layer and the node resumes data transmission process. Similarly, all the intermediate nodes resume their transmission processes. As a result, the source node does not need to retransmit all the data packets during a link failure, increasing the overall throughput of the network.
In what follows, we describe the aforementioned buffering mechanism with the help of an example. Consider the figure 3(a), where node $B$ forwards the data packets of three sources $S_1$, $S_2$ and $S_3$ to three destination nodes $D_1$, $D_2$ and $D_3$, respectively. Figure 3(a) shows the normal situation in which all the data packets are queued at network layer queue of node $B$.

Now, suppose the link between node $B$ and $F$ is broken. As soon as node $B$ detects the link failure, it buffers all the nodes for the destination node $D_2$ in its TQ and continues its normal operation for other destinations. Meanwhile, $B$ starts a local recovery process for destination node $D_2$ and sends a Route Disconnection Notification (RDN) message towards the source node $S_2$. All the intermediate nodes buffer their packets for the destination node $D_2$ in their TQs on receiving RDN message. Node $S_2$ stops its transmission and waits for a reply from node $B$. Figure 3(b) describes this situation. Whenever node $B$ finds a local path $B \rightarrow N \rightarrow D_2$, it sends Route Successful Notification (RSN) control message toward source $S_2$ and resumes its transmission process from transport layer queue. All the intermediate nodes resume their transmission process same way after getting the RSN message. Such a mechanism decreases packet dropping rate and node $S_2$ does not need to transfer...
all the packets. Figure 3(c) shows node B performs normal operation from network layer as well as resumes transmission process from its TQ for destination node \(D_2\).

However, if no local path is found any link, the intermediate node creates another control message called Route Unsuccessful (RUN) message and sends it back to the source node. The source node, on receiving the RUN message, starts a new route discovery process.

4.2. Proposed Energy-efficient Routing Scheme in Network Layer

In MANET, in order to prolong the network lifetime, it is very important to reduce and balance energy consumption throughout the network. This can be achieved by energy-efficient routing. Death of certain critical nodes may lead to the network partition, rendering other live-nodes unreachable.

Let \(\mathcal{K}\) denote the set of all candidate paths for the route, among which the route(s) should be selected. A candidate path \(k\) \((k \in \mathcal{K})\) can be represented by an ordered list of nodes forming the path. That is, the list can be denoted by \([n^k_0, n^k_1, \ldots, n^k_{h_k(k) - 1}, n^k_{h_k(k)}]\), where \(h(k)\) is the hops for path \(k\) and the nodes \(n^k_0\) and \(n^k_{h_k(k)}\) correspond to source and destination, respectively. Let \(\mathcal{N}_k\) denote the set of nodes on path \(k\), excluding source and destination. Note that \(|\mathcal{N}_k| = h(k) - 1\), where \(|X|\) means the number of elements in the set \(X\). To prevent looping and excessive delay, the hop count of a path should be limited. That is, \(h(k) \leq \text{TTL}\) for all \(k\), where \(\text{TTL}\) is the limit on the number of hops (time to live).

Let \(E_{\text{res}}^n\) be the residual battery energy of node \(n\). A threshold \(E_{\text{thr}}\) is used to improve the survivability of nodes. When \(E_{\text{res}}^n < E_{\text{thr}}\), node \(n\) does not participate in any candidate path for other source-destination pair. Let us define the minimal nodal residual energy on path \(k\) as \(mE_{\text{res},k} := \min_{n \in \mathcal{N}_k} E_{\text{res}}^n\). In selecting route among \(\mathcal{K}\) candidates, we can consider a lot of factors. For energy-efficient routing, the most important factors are the minimal nodal residual energy. The survivability of nodes with low battery energy can be largely improved by not to select the candidate path with low minimal nodal residual energy. This, in turn, extends the lifetime of ad hoc network. On the other hand, by selecting the candidate path with small hop count, the network-wide energy consumption can be reduced as well as the packet delay can be shortened. Thus, we define the utility for path \(k\) as a function of these two factors, denoted by \(U(mE_{\text{res},k}, h(k))\).

Now, the routing problem can be formulated as follows.

\[
\begin{align*}
\arg\max_{k \in \mathcal{K}} U(mE_{\text{res},k}, h(k)) \\
\text{s.t.} \quad h(k) \leq \text{TTL}, \quad k \in \mathcal{K}, \\
E_{\text{res}}^n \geq E_{\text{thr}}, \quad n \in \mathcal{N}_k, \quad k \in \mathcal{K}
\end{align*}
\]

Although the solution of (1) is an optimal route, it is hard to implement this algorithm directly in practice. The main reason is that, in order to take account of all possible candidate paths in a distributed manner, all nodes in the network should exchange a huge amount of control information, which is unrealistic. Thus, we propose the routing protocol which is satisfactorily efficient from the practical point of view, even though it is not optimal in the strict sense.

The key operations in the proposed energy-efficient routing scheme are the route discovery/selection and the route maintenance. The route discovery/selection is to find a number of routes between a source-destination pair and two types of control packets, the route request (RREQ) and the route reply (RREP), are used for this purpose. First, the source broadcasts RREQ. RREQ is delivered to the destination and because of the broadcast nature, multiple copies of the RREQ can arrive at the destination. Therefore, the destination can have a number of candidate routes. The destination selects one of the best routes among them. We now describe the operations of the routing scheme in detail.

4.2.1. Route Discovery When a source node has packets to transmit to a specific destination, it initiates the route discovery process by broadcasting RREQ packet to its neighbours. The RREQ packet format is shown in 4. In RREQ packet, the type field indicates type of packet, which is sent over the network. The flag field is used to make synchronization. The reserved field holding with
‘0’ value is applied to ignore the packet. The hop-count field is used to count the number of hops from the source to destination. There are two purposes of the TTL (time to live) field: (i) to limit the number of hops in a path and (ii) to reduce the network-wide broadcast overhead of RREQ packet. In order to identify a route, it uses RREQ-ID field. The originator-ID and destination-ID indicate IP addresses of the source and destination, respectively. The originator sequence number field provides current-sequence number of the route entering the source. The destination-sequence number is used for route-entry pointing to the destination. The $mE_{res}$ indicates the minimum residual nodal energy in a route.

<table>
<thead>
<tr>
<th>Type</th>
<th>J</th>
<th>R</th>
<th>G</th>
<th>U</th>
<th>Reserved</th>
<th>Hopcount</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>J</td>
<td>R</td>
<td>G</td>
<td>U</td>
<td>Reserved</td>
<td>Hopcount</td>
</tr>
<tr>
<td>RREQ-ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source-ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source-Sequence-Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination-ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination-Sequence-Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$mE_{res}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4. RREQ packet format

Every node has a threshold on battery energy, denoted by $E_{th}$, to contribute as an intermediate node of a route. If the battery energy of an intermediate node is less than $E_{th}$, it does not participate in the route discovery process by dropping the corresponding RREQ. While re-broadcasting the modified RREQ by intermediate node, it increments the value of hop-count and decrements the value of TTL. After receiving an RREQ packet, a node checks the TTL value. If it is zero and the node is not destination, the node drops the RREQ. Otherwise, the node decreases the value by one as discussed above.

The proposed routing protocol aims to balance the nodal energy consumption, which can prevent one or some critical nodes from early energy exhaustion that may results network partition. The energy-efficient paths are selected by considering the larger minimal nodal residual energy of the candidate paths. To do so, the source node sets the $mE_{res}$ field (which indicates the minimum residual nodal energy of a path) as the value of its battery energy. When an intermediate node receives the RREQ packet, it compares the $mE_{res}$ in the RREQ with its own residual battery energy, $E_{res}$. If $E_{res}$ is smaller than the value in the $mE_{res}$ field, the value is replaced with $E_{res}$. As a result, the $mE_{res}$ field of RREQ arrived at the destination contains the lowest battery energy value among all the nodes in the corresponding route. This value is called the ‘minimal residual path energy’.

4.2.2. Route Selection The destination makes route decision by selecting multiple appropriate paths. For this purpose, destination collects at maximum $Q$ copies of an RREQ corresponding to a connection request. To do so, the destination starts waiting timer after receiving the first RREQ copy and collects all copies of the RREQ until the timer expires. If it gathers $Q$ copies of an RREQ, it stops collecting even though the timer does not expire.

After collecting one or more RREQ copies, the destination makes route decision. In decision, the route having larger minimal residual path energy gets higher priority. On the other hand, the route having smaller hop-count is favorable because it may provide shorter delay. There can be many utility functions taking account of these two factors. Among them, we use the route utility (4) for its simplicity.

$$U(mE_{res,k}, h(k)) = \frac{mE_{res,k}}{HC_k}, \quad k = 0, 1, \ldots, K;$$

where, $mE_{res,k}$ is minimal residual path energy of $k$th routing path and $K (K \leq Q)$ is the number of collected RREQ copies. And $HC_k$ is the hop count of $k$th RREQ copy. $U(mE_{res,k}, h(k))$ is the
route utility of $k$th routing path. The destination sorts all utilities in a descending order. Then, it selects first $P$ paths among $K$ candidates. The order of a selected path becomes the priority of the corresponding path.

The destination sends an RREP packet corresponding to the path with highest priority. It is noted that only destination node is allowed to send RREP. An RREP contains the accumulated route record obtained from the corresponding RREQ copy. If route discovery process is successfully accomplished, the source node receives RREP for an RREQ.

4.2.3. Route Maintenance

The route failure can occur during data transmission. The proposed routing protocol has two types of route recovery mechanisms. The first one is local recovery, whereas the other one is end-to-end recovery. Let us consider a link between an upstream node $A$ and a downstream node $B$, which are intermediate nodes on the route between source and destination. The link can be failed because of the energy exhaustion in nodes, mobility, or the degradation of wireless channel condition.

If node $A$ cannot receive ACK from node $B$ after it retransmits a packet $N_{\text{retx}}$ times ($N_{\text{retx}}$ is predefined), the node $A$ starts local recovery process as discussed in Subsection 4.1. In local recovery, as soon as node $A$ detects a link failure, it buffers all the packets for the destination node in its TQ and continues its normal operation for other destinations. Meanwhile, $A$ starts a local recovery process for destination node and sends a RDN message towards the source node. All the intermediate nodes buffer their packets for the destination node in their TQs on receiving this RDN message. The source node stops its transmission and waits for a reply from node $A$.

Node $A$ tries to find another path from node $A$ to node $B$, which should contain only one additional intermediate node. To do so, node $A$ gathers neighbor information, determines a candidate for the additional intermediate node (say, node $C$), and transmits a local route recovery control packet to node $C$. Then node $C$ transmits a modified recovery packet to node $B$. As a result, a detour $A$–$C$–$B$ is made. It is noted that, the resulting end-to-end route is valid when its hop count is not greater than TTL. Whenever node $A$ finds a partial path, it sends RSN message toward source node and resumes its transmission process from transport layer queue. All the intermediate nodes resume their transmission process same way after getting the RSN message. Such mechanism decreases packet dropping rate and source node does not need to transfer all the packets. When the local recovery is impossible, node $A$ sends a RUN message to the source. After receiving the RUN packet, the source starts to the rerouting process. This is called the ‘end-to-end recovery’.

4.3. Congestion Detection and Control Mechanism in the Proposed Cross Layer Design

Our approach uses congestion-aware data delivery mechanism so that nodes in the network can easily identify the congestion level of the network and take appropriate action. At each intermediate node, we measure the congestion level and piggybacks that information toward the source node so that appropriate control actions can be taken in time. We use two bits control flag in both data packets and acknowledgement packets, referred to as Congestion Notification (CN) flag. Every node in an active route sets this flag when they forward packets. The value of the CN flag detects the congestion level of the network according to table I. From the value of the CN flag, the neighborhood nodes can easily be informed about the congestion status of the network and they can take proper actions to handle congestion. What follows, we describe how a node detects the congestion level and assign the value of CN and how the congestion is controlled based on the value of CN flag.

<table>
<thead>
<tr>
<th>Value of CN</th>
<th>Congestion level</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Lightly loaded</td>
</tr>
<tr>
<td>01</td>
<td>Loaded</td>
</tr>
<tr>
<td>10</td>
<td>Heavily Loaded</td>
</tr>
<tr>
<td>11</td>
<td>Congested</td>
</tr>
</tbody>
</table>

Table I. Congestion notification
4.3.1. Detection of congestion level Based on queue occupancy, here we use an early congestion detection technique by which a node can detect the current congestion status. We use minimum and maximum thresholds, $Q_{\text{min}}$ and $Q_{\text{max}}$, respectively, for queue occupancy at any node as follows:

\[
Q_{\text{min}} = l \times Q_{\text{size}},
\]

\[
Q_{\text{max}} = h \times Q_{\text{size}},
\]

where, $l$ and $h$ are two control parameters; in our simulation, we set $l = 0.5$ and $h = 0.9$, respectively. If the queue length of a node is less than $Q_{\text{min}}$, then we can say the network is lightly loaded, e.g., queue occupancy is less than 50%; if the queue length is greater than $Q_{\text{min}}$ but less than $Q_{\text{max}}$, then it is operating in the safe region; and, if the queue length is greater than $Q_{\text{max}}$, the node is considered as congested. Even though the above thresholds help to identify congestive or non-congestive states, they don’t protect nodes moving from non-congestive state to congestive one. In support of implementing congestion-aware data delivery mechanism, we introduce a warning threshold parameter $Q_{\text{warn}}$, defined as follows:

\[
Q_{\text{warn}} = w \times Q_{\text{size}},
\]

where, $w$ is a weight factor and in our simulation we choose $w = 0.8$.

We then calculate average queue occupancy of a node every after a certain interval using Exponentially Weighted Moving Average (EWMA) formulae as follows:

\[
Q_{\text{avg}} = (1 - \alpha) \times Q_{\text{avg}} + Q_{\text{curr}} \times \alpha
\]

where $\alpha$ is a weight factor and $Q_{\text{curr}}$ is the current queue size. Now, based on the value of $Q_{\text{avg}}$, we determine the value of CN flags as follows:

- if $Q_{\text{avg}} < Q_{\text{min}}$ then CN = 00
- if $Q_{\text{avg}} \geq Q_{\text{min}}$ and $Q_{\text{avg}} < Q_{\text{warn}}$ then CN = 01
- if $Q_{\text{avg}} \geq Q_{\text{warn}}$ and $Q_{\text{avg}} \leq Q_{\text{max}}$ then CN = 10
- if $Q_{\text{avg}} > Q_{\text{max}}$ then CN = 11

4.3.2. Congestion Control Mechanism In our proposed RCECD, a node takes proper actions to control the congestion according to the values of CN bits. If the value of CN flag at a node is ‘00’, it assumes the network is lightly loaded. In such case, the node performs its normal operations. It allows the other nodes of the network to transmit packets through it. So the node accepts new RREQ messages from new source and rebroadcasts to create new routes through it. When the value of CN is ‘01’, it discards new RREQ messages, allowing no new route through it in order to avoid any future congestive states; however, in this case, the sources of the existing routes may increase their traffic rates passing through this node. If the value of the CN is ‘10’, the network becomes heavily loaded. So, further increasing of data arrival rates from source nodes will lead the network to fall into congestive state. In this case, the node generates a new control message called ALERT message and sends back toward every source node so that they do not increase the data forwarding rates. Thus, our proposed RCECD controls the network in ahead of time and implements a congestion-aware reliable data delivery mechanism. But if a node detects the value of CN is ‘11’, this means the network has already fallen into congestive state and it then starts buffering data packets in TQ and stops forwarding packets afterwards. The table II shows the operations taken by a node for different congestive states.

5. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposed RCECD mechanism in network simulator v-2.34 [29] and compare the simulation results with that of AODV-BBS [5], SMR [7], THR [11] and MTEC [21]. The results of our simulation state that the RCECD outperforms the other protocols.
Table II. Actions taken by a node to control the congestion

<table>
<thead>
<tr>
<th>Congestion Level</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lightly loaded</td>
<td>Normal operation - no change</td>
</tr>
<tr>
<td>Loaded</td>
<td>Stop forwarding any RREQ message</td>
</tr>
<tr>
<td>Heavily loaded</td>
<td>Send ALERT message to all sources</td>
</tr>
<tr>
<td>Congested</td>
<td>Start buffering packets</td>
</tr>
</tbody>
</table>

5.1. Simulation Environment

In our simulation, we consider a square area of size 1000 × 1000m², where 100 mobile nodes are deployed randomly. The simulation time is set to 400 seconds. Each node has the transmission range of 250m initially had 100 J of energy. The source nodes of our network generate constant bit rate (CBR) data streams. This helps to measure performance for various traffic load at each mobile node. The size of each data packet is 1500 bytes, link bandwidth is kept at 11 Mbps and the underlying transport and MAC layer protocols are UDP and IEEE 802.11 DCF, respectively. The table III summarizes the simulation parameters. For each data point in the graphs, we take the average of 10 simulation runs that helps us to study the steady state behavior of the protocols.

5.2. Performance Metrics

We use five performance metrics to compare the results of AODV-BBS, SMR, MTEC, THR, and RCECD. Those metrics are as follows:

- **Packet delivery ratio** is measured as the ratio of the total number of received data packets by all the destination nodes to the total number of generated data packets by all the source nodes in the network.

Table III. Simulation parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network area</td>
<td>1000m x 1000m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>100</td>
</tr>
<tr>
<td>Deployment type</td>
<td>Random</td>
</tr>
<tr>
<td>Number of sources</td>
<td>20</td>
</tr>
<tr>
<td>Node movement model</td>
<td>Random waypoint</td>
</tr>
<tr>
<td>Transmission range</td>
<td>250m</td>
</tr>
<tr>
<td>Transport layer protocol</td>
<td>UDP</td>
</tr>
<tr>
<td>MAC layer protocol</td>
<td>IEEE 802.11 DCF</td>
</tr>
<tr>
<td>Control packet size</td>
<td>100 bits</td>
</tr>
<tr>
<td>TTL limit</td>
<td>5</td>
</tr>
<tr>
<td>Number of retransmission limit, N_{retx}</td>
<td>3</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>11 Mbps</td>
</tr>
<tr>
<td>Node initial energy</td>
<td>100 J</td>
</tr>
<tr>
<td>Threshold on battery energy, E_{th}</td>
<td>5 J</td>
</tr>
<tr>
<td>Data size</td>
<td>2 Mbytes/flow</td>
</tr>
<tr>
<td>Data packet size</td>
<td>1500 bytes</td>
</tr>
<tr>
<td>Data packet generation rate</td>
<td>0.1 1.0 Mbps</td>
</tr>
<tr>
<td>Propagation model</td>
<td>Free Space</td>
</tr>
<tr>
<td>Weight factor, ( \alpha )</td>
<td>0.2</td>
</tr>
<tr>
<td>Simulation time</td>
<td>400 seconds</td>
</tr>
</tbody>
</table>
Average end-to-end packet delay is measured as the average time in ms required by all the data packets that are received by the destination nodes.

Throughput is measured as the average amount of data bits received per unit time by all the destination nodes in the network.

Energy efficiency is measured as the number of successful data packets received by the destination per joule.

Normalized routing overhead is measured as the number of control packets generated during the simulation period for each successfully delivered data packet.

5.3. Impact of Varying Traffic Loads

In this section, we study the impact of different data generation rates or traffic loads on the performances of the protocols in terms of the above metrics. The data generation rates at source nodes are varied from 0.1 ~ 1.0 Mbps. For this experiment, we fix the mobility speed of each node at 2 m/s within the network.

Fig. 5 shows the average end-to-end packet delay performances of the protocols for various traffic loads. It can be seen the proposed RCECD approach had a lower average end-to-end delay than THR, MTEC, SMR and AODV-BBS by 2.511% respectively. AODV-BBS and SMR experience much higher delay than others due to high latency of maintaining multiple alternate routes. MTEC has longer delay than THR and RCECD since it can not handle the congestion. However, our RCECD handles congestion proactively. This strategy helps RCECD nodes to operate in safe mode (i.e., loaded state) most of the time and thus decreases the queuing delays of the packets at the intermediate nodes, which in turn decreases the end-to-end packet delivery delay a lot. In addition, the proposed energy-efficient routing protocol considers hop count as one of the routing metrics that plays an important role to reduce the delay in the network.

Figure 5. Average end-to-end delay for different traffic loads

Fig. 6 shows the packet delivery ratios with different data generation rates. The simulation results indicate that our proposed RCECD mechanism provided higher packet delivery ratios than other protocols by 4%, 7%, 9% and 12% respectively. It can be seen from Fig. 6 that the packet delivery ratio for all the protocols decreases with increased source data traffic loads due to congestion or packet drops at intermediate nodes. Still the proposed approach performs well as compared to others since it can take appropriate control actions in time when congestion happens. In addition, the proposed energy-efficient routing in the network layer can balance the energy consumption throughout the network which helps to improve the packet delivery ratio. Furthermore, the buffering mechanism of RCECD reduces the packet drops a lot and thus increases the packet delivery ratio.

Based on the analysis and discussion above, we can see that RCECD produced a lower average end-to-end delay and a higher delivery ratio of packets than the other routing protocols. Therefore,
RCECD provided better throughput than the other methods, as shown in Fig. 7. We can see that the throughput of RCECD was 1.9 Mbps, which was better than the other protocols since it ensures higher number of packet delivery at the destination within minimum end-to-end delay. The throughput of the protocols increase as the traffic load increases but it starts decreasing at around 0.7 Mbps traffic load, where the network reaches at saturation condition. Still the proposed approach performs better than others.

Fig. 8 shows the comparison of average network energy efficiency of RCECD scheme with other protocols in terms of various traffic loads. From this plot, we can observe that, the network energy efficiency of the proposed scheme outperforms the other protocols, although the energy efficiency of our proposed scheme reduces when the traffic loads exceed 0.7 Mbps. The reason of the improvement of energy efficiency in the proposed scheme is because of the utilization of energy-efficient routing in network layer and the buffering technique in the transport layer along with multilevel congestion control mechanisms.

Figure 9 shows the normalized routing overhead of the studied protocols. The AODV-BBS has high latency of route discovery process for keeping multiple routes for same destination and generates large amount of control packets. During link failure, SMR needs a fresh route discovery process, producing a large number of control packets. In case of MTEC, the overhead is less
Figure 8. Energy-efficiency for different traffic loads

Figure 9. Normalized routing overhead for different traffic loads

than AODV-BBS and SMR due to energy-efficiency and cross-layer design. The THR can handle link failure but can not cope up with network congestion as traffic load increases and local route discovery process starts on the failure of links. On the other hand, our RCECD neither maintains any alternative routes nor it balances loads among multiple routes; rather, it uses multilevel congestion control mechanism and local packet buffering at transport layer for the period of local route discovery to mitigate link failures and congestion in the network. As a result, it generates minimum number of control packets and thus provides with the least normalized routing overhead among the studied protocols. In addition, the proposed energy-efficient routing protocol reduces the broadcasting messages and protects the early drain of the critical nodes that also helps to reduce the link failure and results in less routing overhead.

5.4. Impact of varying route failure rates

In this section, we evaluate the impact of varying route failure rates on the performances of the studied protocols, keeping the packets generation rate constant 0.7 Mbps. We vary the route failure rates from 1 ~ 10 routes/sec.
The graph in Fig. 10 states that the packet delivery ratio for all protocols decreases sharply with the increasing route failure rates. This happens because the failure of routes increases the number of packet drops. Since our proposed RCECD scheme exploits the multilevel congestion control, packet buffering, and energy-efficient routing mechanisms, it is capable of addressing route failures more effectively and saves packets from dropping and thus it can increase the packet delivery ratio compared to other protocols.

Fig. 11 shows the end-to-end packet delivery delay with the increased route failure rates for all the protocols. We can see that SMR experiences the longest end-to-end packet delivery delay. MTEC has lower end-to-end delay than SMR but higher than others protocols. AODV-BBS uses an alternative route whenever a link failure is detected and thus it decreases the delay than SMR and MTEC. The local route discovery-assisted packet buffering mechanism in RCECD helps it to handle route failures more efficiently than others. The number of retransmissions required on the failure of routes decreases a lot and thus it reduces the traveling time of data packets.

Figure 12 shows the throughput performances of the studied protocols. The throughput decreases for all the protocols as the route failure rate increases. Because of high latency multiple paths, AODV-BBS and SMR provide lower throughput. MTEC provides higher results since it uses cross
layer design. RCECD has the higher throughput over the four other protocols since it has efficient route failure handling mechanism, and energy-efficient routing protocol.

Figure 13 shows the average energy efficiency of all the protocols in terms of various route failure rate. From this plot, we can observe that, the energy-efficiency of the proposed RCECD scheme outperforms the other approaches. With the increment of route failure, the value of the number of control packets generated increases in the network, which consumes more energy and frequent route maintenance is needed. Because of packet buffering capabilities, efficient congestion control mechanism as well as energy-efficient routing protocol, RCECD provides better result than other protocols.

Figure 14 shows the normalized routing overhead in terms of route failure rates. It can be seen that the RCECD provided low overhead than other protocols because of its packet buffering capabilities and local recovery process as well as utilizing the energy-efficient routing protocol.

6. CONCLUSION

In this paper, we propose a route failure and congestion-aware energy-efficient cross layer design for the MANETs to provide reliable and timely data delivery. The proposed design combines the
transport and network layer. In the transport layer, the concept of local packet buffering is proposed to store incoming data packets, when needed. The nodes on an active route buffer incoming packets at their local TQs during link failure and, on finding a new path, resume their transmissions. As a result, packet dropping rate of the network decreases. In the network layer, an energy-efficient routing protocol is proposed that can balance the energy consumption throughout the network as well as reduces energy consumption by limiting broadcast messages. In addition, we employ a multilevel congestion detection and control mechanism at the source and intermediate nodes that can judiciously take the most appropriate decision for congestion control in the network proactively. The simulation results showed that the proposed cross layer design provided better packet delivery ratio and throughput than existing protocols. It also offered better energy-efficient path during data transmission, decreased end-to-end packet delivery delay as well as routing overhead than state-of-the-art protocols. In future, we will consider the multichannel environments of MANETs as well as the mobility issues in more details which are the most challenging factors. The proposed protocol can be extended by combining the MAC layer with the Network Layer. We will focus on, with the presence of host’s mobility, how to conserve as well as adjust power energy dynamically to maintain the route’s connectivity and restarting the route discovery periodically to find a new route with better energy efficiency.

ACKNOWLEDGEMENT

This work was supported by the Research Center of College of Computer and Information Sciences, King Saud University, Project No: RC120915. The authors are grateful for this support

REFERENCES


Weng CC, Chen CW, Chen PY, Chang KC. Design of an energy-efficient cross-layer protocol for mobile ad hoc networks. IET Communications 2013; 7(3):217–228.

Kumar RS, Kamalakannan P. A review and design study of cross layer scheme based algorithm to reduce the link break in manets. Pattern Recognition, Informatics and Medical Engineering (PRIME), 2013 International Conference on, IEEE, 2013; 139–143.


