Minus HELLO: Minus Hello Protocols for Energy Preservation in Mobile Ad-hoc networks

Anuradha Banerjee · Abu Sufian · Paramartha Dutta

Received: date / Accepted: date

Abstract In mobile ad-hoc networks, nodes have to transmit HELLO or Route Request messages at regular intervals, and all nodes residing within its radio range, reply with an acknowledgment message informing their node identifier, current location and radio-range. Transmitting these messages consume a significant amount of battery power in nodes, especially when the set of down-link neighbors do not change over time and radio-range of the sender node is large. The present article focuses on this aspect and tries to reduce number of HELLO messages in existing state-of-art protocols. Also, it shortens radio-ranges of nodes whenever possible. Simulation results show that the average lifetime of nodes greatly increases in proposed minus HELLO embedded routing protocols along with a great increase in network throughput. Also, the required number of route re-discovery reduces.

Keywords Ad-hoc networks · AODV · Energy Preservation · green communication · MANET · Minus HELLO · Reactive Routing

1 Introduction

An ad-hoc network or simply MANET is an infrastructure-less network consisting of only some mobile nodes that move freely in any direction [1], [2]. These networks can be deployed in emergency situations like war, natural disaster, etc [3]. Battery powered nodes act as endpoints or routers to selflessly forward packets in a multi-hop environment [4], [5]. Therefore, energy efficiency in every node is crucial to preserve battery power of nodes and increase their lifetime [6], [7], [8].

A. Banerjee is with Dept. of Computer Applications
Kayani Govt. Engg. College, WB, India

A. Sufian is with Dept. of Computer Science
University of Gour Banga, WB, India. Tel.: +91-9732905753
E-mail: sufian.csa@gmail.com

P. Dutta is with Dept. of Computer and System Sciences
Visva-Bharati University, WB, India
1.1 Contributions of this proposed scheme

i) The present article proposes a novel idea of reforming routing protocols so that they can work with very less HELLO or Route Request (RREQ) messages. Minus HELLO (-HELLO) emphasizes that being informed about downlink neighborhood is absolutely unnecessary until and unless a node participates in a communication session as source or router. For simplicity of the representation, we shall refer to Minus HELLO as -HELLO in the rest of the article.

ii) A general framework of -HELLO version of state-of-art protocols is presented with mathematical illustration as case studies.

iii) Message formats of state-of-art protocols have been re-designed to contain certain additional attributes to overcome the absence of HELLO.

iv) It has been shown in our paper that hidden and exposed terminal problems can be resolved with very less HELLO messages. Therefore, robustness of the network does not suffer.

v) Detailed simulation results emphasize that the communication protocols with very less HELLO messages, save a lot of energy with a significant increase in network throughput.

1.2 Organization of the article

Organization of the present article is as follows. Section 2 deals with a brief description of various routing protocols in MANETs, including the proactive, reactive and energy efficient ones. General improvement produced by -HELLO version protocols is highlighted in section 3. Section 4 explains -HELLO embedded protocols as case studies, such as -HELLO versions of the protocols AODV, MMBCR, MRPC, MTPR and MFR. Here we have omitted proactive routing protocols because they are not suitable for large networks that is, when the number of nodes is high. AODV and MFR are two state-of-the-art representatives of reactive routing protocols, whereas MMBCR, MTPR and MRPC are three popular representatives of energy efficient protocols. Section 5 presents the simulation results while section 6 concludes the paper.

2 Existing Routing Protocols for Communications

The literature of MANETs is rich in proactive, reactive and energy-efficient protocols [9]. Destination-sequenced Distance Vector (DSDV) [10], Cluster-based Gateway Switch Routing (CGSR), Global State Routing (GSR) [11], The wireless routing protocol (WRP) [12], Fisheye state routing (FSR) [13] etc. are state-of-the-art proactive protocols. These instruct the nodes to store route information to every other node in the network. Hence, a regular update of routing tables is required, consuming huge battery power as well as bandwidth.

Among reactive routing protocols, ad hoc on-demand distance vector (AODV) [14], dynamic source routing (DSR) [15], flow-oriented routing protocol (FORP) [16], The Temporally Ordered Routing Algorithm (TORA) [17], The operation of location aided routing (LAR) [18], Most forward with fixed radius or MFR [19] etc. have become standard. Here routes are discovered on-demand through a RREQ,
route-reply(RREP) cycle. A RREQ packet reaches the destination through multiple paths. Among them, one is elected by the destination according to the routing protocol, and sent to the source through a RREP packet, so that the source node can start sending data packets to destination through the chosen best path.

For energy conservation schemes, variety of scheme comes with different energy saving strategies. Some are based on the concept of adjusting radio range of senders in each hop[14]. Maximum residual packet capacity (MRPC)[20] selects the path that has maximum number of packets to transmit. This computation is based on the residual energy of nodes involved in the path. Minimum battery cost routing (MBCR)[21] aims to find a route with maximum remaining battery capacity. The cost of a node is \((1/residual\_battery\_power)\), and the cost of a route is summation of the costs of all its nodes. The route with minimum cost is elected for communication. Min-max battery cost routing (MMBCR)[21] assigns a performance index of a route with minimum of battery powers of all nodes in the route. Among multiple routes through which a packet arrives at the destination, the one with maximum performance index, is chosen for communication. Minimum Transmission Power Routing (MTPR)[22] selects the path with minimum transmission power, for transferring data packets. Computation of minimum transmission power is done according to Frii’s transmission equation[22]. An energy harvesting technique is proposed in[22] where transmitter changes its location to identify better energy harvesting spots and this harvesting energy is utilized for actual data transmission by the current transmitter. Some schemes proposed sleeping strategies for saving energy of nodes. In[24] exhausting nodes are allowed to go to sleep for a pre-defined time period, after which they wake up and resume communications.

Some routing schemes other than above three also has existence in literature. Flow oriented routing protocol, or FORP[9] is a stable path routing protocol that produces comparatively stable paths compare to earlier protocols. In FESC[25] a stable single hop clustering scheme has proposed. Here more battery powered but less mobile nodes are elect for cluster head and all other nodes directly connect to nearest cluster head. In SR-MQMR[26], the authors try to increase stability and energy efficiency through multipath routing. Associativity Based Routing (ABR) protocol[27] where beacons are exchanged periodically between neighbors.

3 GENERAL IMPROVEMENT PRODUCED BY -HELLO VERSION PROTOCOLS

3.1 Background and basic idea of the scheme

In MANET, nodes regularly broadcast HELLO messages within their respective radio-ranges, to gain information about their one hop downlink neighborhood. All nodes lying within radio-circle of sender of the HELLO message, reply with acknowledgment or ACK informing their unique identification number, location, radio-range etc. HELLO messages are useful from communication perspective because they enable a node to be aware of available links, among which one is chosen as per performance metric of the underlying protocol. Also hidden and exposed terminal problems are tackled in MANETs with the help of HELLO messages. But this HELLO dependency has disadvantage too. HELLO and ACK messages are
exchanged by each node at regular intervals even when the node is not initiating a communication. This eats up huge energy in nodes and reduces their lifetime [28].

Our present article focuses on this particular problem. It aims at redesigning state-of-the-art representatives of routing protocols so that they can work with very less HELLO messages without losing robustness of the networks. -HELLO points out the fact that these information are irrelevant until and unless a communication request arrives at the node. In -HELLO, neighborhood information is collected during the broadcasting of RREQ. Eliminating irrelevant HELLO and ACK messages contribute to save a huge amount of energy in the network. This will improve the average lifetime of nodes. As a result, link breakages due to node battery exhaustion will be reduced up to a great extent. Therefore, number of RREQ messages injected into the network, will be greatly reduced for -HELLO version protocols.

3.2 Mathematical Analysis of -HELLO version of protocols

-HELLO embedded protocols particularly reduced HELLO messages from reactive and energy-efficient routing protocols. It focuses on the fact that information about neighbors of a node is typically required during route discovery, that is, at the beginning of a communication session when route to a specific destination is to be found out. For that purpose, size of RREQ messages in various protocols increase a bit. Case study in the next section show how HELLO messages can be eliminated or reduced for performance improvement in various standard routing protocols in MANETs, like AODV, MBCR, MTPR etc. Below we mathematically demonstrate improvements that can be produced by -HELLO versions of protocols in respect of lifetime, throughput, delay etc. Let us denote by $ln(i)$ the link between two nodes $n_i$ and $n_{i+1}$. The status of each node is either up or down. If a node $n_i$ is operational, then its status will be up; otherwise down. According to the study of discharge curve of batteries heavily used in MANETs, at least 40% of total battery power is required to remain in operational condition [29]. Therefore, if $max_{eng}(i)$ and $res_{eng}(i,t)$ denote maximum and residual energy of node $n_i$ at time $t$, then $n_i$ will be up provided condition in equation (1) is true.

$$res_{eng}(i,t) > (0.4 \times max_{eng}(i))$$

$RT(n_i,n_{i+1})$ is a random variable which indicates liveliness of $ln(i)$ from the perspective of mobility. It indicates potential communication capability of the link from $n_i$ to $n_{i+1}$. It is 1 if $n_{i+1}$ is in radio-range of $n_i$, otherwise it is 0. Then, probability that a route $ROUTE_{(s,d)}$ from $n_s$ to $n_d$ is live, is denoted by $P(ROUTE_{(s,d)})$, and $P(n_i)$ denotes probability of node $n_i$ is live. Mathematical expression of this appears [30] in equation(2).

$$P(ROUTE_{(s,d)}) = \prod_{0}^{m} P(n_i) \prod_{0}^{m} RT(n_i, n_{i+1})$$

-HELLO version of protocols cannot improve mobility oriented stability of links, that is, $RT(n_i,n_{i+1})$, but it greatly enhances $P(n_i)$ for all $i$ s.t. $0 < i < m$, as shown in the following lemmas.
Lemma 1: -HELLO embedded protocols greatly enhance lifetime of nodes.

Proof: Assume that the minimum, maximum and average values of minimum receive powers of nodes in the network are given by min_min_rvc, max_min_rcv and avg_min_rcv respectively. Hence average transmission power $\text{avg}_\text{trans}(i)$ of $n_i$ to process a call, is formulated in equation (3).

$$\text{avg}_\text{trans}(i) = \text{avg}_\text{min}_\text{rvc}(0 + R_i^2)/2C$$

where $R_i$ is radio-range of $n_i$ and $C$ is a constant depending on medium.

Let $L(i)$ be lifetime of $n_i$ in HELLO version protocols, $y$ be the number of HELLO messages transmitted by each node in the network per unit time and $\text{broad}(i)$ is the unit of energy required to broadcast a message. Total number of HELLO messages transmitted by $n_i$ throughout its lifetime, is given by $(y \times L(i))$ and the corresponding energy required to broadcast, is $(y \times L(i) \times \text{broad}(i))$ units. Amount of energy units, $n_i$ consumes for processing calls throughout its lifetime, is $(\text{rt}(i) \times L(i) \times \text{avg}_\text{trans}(i))$ units where $\text{rt}(i)$ denotes the total number of message packets it transmitted. Throughout lifetime of a node $n_i$, it can use only $0.6 \times \text{max}_\text{eng}(i)$ amount of energy. Hence,

$$L(i) = (0.6 \times \text{max}_\text{eng}(i))/(y \times \text{broad}(i)) + \text{rt}(i) \times \text{avg}_\text{trans}(i)$$

(4)

As far as -HELLO version protocols are concerned, let $L'(i)$ be lifetime of $n_i$, and it can be mathematically formulated in equation (5).

$$L'(i) = (0.6 \times \text{max}_\text{eng}(i))/(\text{rt}(i) \times \text{avg}_\text{trans}(i))$$

(5)

Improvement in lifetime produced by -HELLO version protocols, is calculated by $(L'(i) - L(i))$ and it is given in equation(6).

$$L'(i) - L(i) = (0.6 \times \text{max}_\text{eng}(i))\{1/(\text{rt}(i) \times \text{avg}_\text{trans}(i)) - 1/(y \text{broad}(i) + \text{rt}(i) \times \text{avg}_\text{trans}(i))\}$$

(6)

i.e. $L'(i) - L(i) > 0$

So, improvement is produced in terms of lifetime. Without any loss of generality we can assume that, in the route $\text{ROUTE}_{s,d}$ from $n_s$ to $n_d$, $n_s$ is the node with minimum residual lifetime. Assuming each packet requires $\text{tme}$ time duration to reach from source to destination through this path. If $\text{pac}$ is the number of packets are to be transferred through this route, then each node in $\text{ROUTE}_{s,d}$ should be live for at least $(\text{tme} \times \text{pac})$ time duration to avoid route-breakage due to battery exhaustion in nodes. If $L'(i) \geq (\text{tme} \times \text{pac})$ and $L(i) < (\text{tme} \times \text{pac})$, then converting a protocol to its -HELLO version will reduce number of route re-discovery sessions in the network. A large number of such route request packets will not require to transfer, which substantially reduce node lifetime. Therefore, it is proved that -HELLO version of protocols greatly enhance lifetime of nodes.

Lemma 2: -HELLO version protocols reduce average waiting time of a packet in message queues of nodes.

Proof: Let call arrival and departure rates at $n_i$ in -HELLO versions protocols are given by $\lambda(i)$ and $\mu'(i)$, and for classical protocols $\lambda(i)$ and $\mu(i)$. Then, from
the Little’s law, average waiting time of a call forwarding request at $n_i$ in -HELLO version of protocols is denoted by $\text{avg\_wait}_{(-\text{HELLO})}(i)$ and defined in equation (7).

$$\text{avg\_wait}_{(-\text{HELLO})}(i) = \lambda(i)/\{\mu(i) \times (\mu'(i) - \lambda(i))\}$$  \hspace{1cm} (7)

Similarly, average waiting time of a call forwarding request at $n_i$ in the classical protocols is denoted by $\text{avg\_wait}(i)$ and defined in equation (8).

$$\text{avg\_wait}(i) = \lambda(i)/\{\mu(i)(\mu(i) - \lambda(i))\}$$  \hspace{1cm} (8)

Call arrival rates increase along with increase of route rediscovery session. Hence, $\lambda(i) > \lambda'(i)$. Let $\lambda(i) = \lambda'(i) + \Delta\lambda; s.t.\Delta\lambda > 0 \hspace{1cm} (9)$

But call departure rate remains same, that is, $\mu'(i) = \mu(i)$, because forwarding capacity of nodes do not change.

Then, $\text{avg\_wait}(i) - \text{avg\_wait}_{(-\text{HELLO})}(i) = F1(i)/F2(i)$

Where $F1(i) = (\lambda'(i) + \Delta\lambda)(\mu'(i) - \lambda'(i)) - \lambda'(i)(\mu'(i) - \lambda'(i)) + \Delta\lambda)$

i.e. $F1(i) = \Delta\lambda\mu'(i)$

$F2(i) = \mu'(i)(\mu'(i) - \lambda'(i) - \Delta\lambda)(\mu'(i) - \lambda'(i))$ So,

$$\text{avg\_wait}(i) - \text{avg\_wait}_{(-\text{HELLO})}(i) = \Delta\lambda/(\mu'(i) - \lambda'(i) - \Delta\lambda(\mu'(i) - \lambda'(i))) \hspace{1cm} (10)$$

Hence, $(\text{avg\_wait}(i) - \text{avg\_wait}_{(-\text{HELLO})}(i)) > 0$

**Lemma 3:** A node acting according to -HELLO version of protocol, produces higher network throughput than classical protocol.

**Proof:** It has already been mentioned that in the classical versions of protocols, call arrival rates increase due to increase in number of route rediscovery sessions, message contention and message collision. Hence, $\lambda(i) > \lambda'(i)$.

From Little’s law, average number of message forwarding requests in message queue of a node $n_i$ with message queue size $mq(i)$ is denoted by $\text{avg\_req}(i)$ and defined in equation (11). Similarly, average number of message forwarding requests in message queue of the same node in -HELLO embedded protocol is denoted by $\text{avg\_req}_{(-\text{HELLO})}(i)$ and defined in equation (12).

$$\text{avg\_req}(i) = \lambda^2(i)/\{\mu(i)(\mu(i) - \lambda(i))\}$$  \hspace{1cm} (11)

$$\text{avg\_req}_{(-\text{HELLO})}(i) = (\lambda')^2(i)/\{\mu'(i)(\mu'(i) - \lambda'(i))\}$$  \hspace{1cm} (12)

It already assumed $\mu(i) = \mu'(i)$. Therefore,

$$\text{avg\_req}(i) - \text{avg\_req}_{(-\text{HELLO})}(i) = F1'(i)/F2'(i) \hspace{1cm} (13)$$

Where, $F1'(i) = \lambda^2(\mu(i) - \lambda'(i)) + \Delta\lambda\lambda'(i)(2\mu(i) - \lambda'(i))$

Hence, $F1'(i) > 0$ and $F2'(i) = F2(i)$. So, $\text{avg\_req}(i) > \text{avg\_req}_{(-\text{HELLO})}(i)$.

From the point of view of packet loss, following different cases arise where classical and -HELLO version protocols compete. We inspect the cases individually and prove that our protocols perform better.
Case-1: Packet loss in -HELLO version is higher than the classical versions. Determination of possible conditions for this case is shown below.

Packet overflow is taking place in both classical and -HELLO versions. Therefore, average number of message requests in both is higher than message queue capacity \( mq(i) \) of \( n_i \). Hence,

\[
\text{avg}_i \text{req}(i) - mq(i) = k_1; \ s.t. k_1 > 0 \quad (14)
\]
\[
\text{avg}_i \text{req}_{-HELLO}(i) - mq(i) = k_2; \ s.t. k_2 > 0 \quad (15)
\]

Approximate packet loss in \( n_i \) in classical protocols and -HELLO version are given by \( k_1 \) and \( k_2 \) respectively, such that \( k_2 > k_1 \), as per assumption in the case. Subtracting equation (14) from (15), we get equation (16)

\[
\text{avg}_i \text{req}_{-HELLO}(i) - \text{avg}_i \text{req}(i) = k_2 - k_1 \quad (16)
\]

But here right side is greater than zero whereas left side is less than zero, which is not possible. Therefore, claim in the statement of case-1 stands false.

Case-2: -HELLO version suffers from packet loss while classical is free from packet loss. The situation can be mathematically modeled in equation (17) and in equation (15) as below:

\[
\text{avg}_i \text{req}(i) - mq(i) = -k_1; k_1 > 0 \quad (17)
\]

Subtracting equation (15) from (17) we get,

\[
\text{avg}_i \text{req}(i) - \text{avg}_i \text{req}_{-HELLO}(i) = -k_1 - k_2 \quad (18)
\]

Here also claim in the statement of case-2 stands false as case-1.

Case-3: Classical versions suffers from packet loss while -HELLO version is free from packet loss. The situation can be mathematically modeled in equation (14) and (19).

\[
\text{avg}_i \text{req}_{-HELLO}(i) - mq(i) = -k_2; k_2 > 0 \quad (19)
\]

Subtracting equation (19) from (14) we get,

\[
\text{avg}_i \text{req}(i) - \text{avg}_i \text{req}_{-HELLO}(i) = k_1 + k_2 \quad (20)
\]

Both sides of the equation (24) are positive, so, case 3 is quite possible.

Case-4: The loss of packets in -HELLO version is smaller than the classical versions. This situation is also formulated as in (14) and (15). But \( k_2 < k_1 \), as per assumption in this case. With the help of previous equations it can be seen that both left and right side of (18) are less than zero, which is quite possible for this case. Hence, the claim that appears in statement of this case is true.

Case-5: The loss of packets in -HELLO version is exactly same as in classical versions. But here, \( k_1 = k_2 \) because packet loss of \( n_i \) in both classical and -HELLO version are same. Therefore, right side of the equation(18) become zero while left side non zero, which is not possible. So, statement in the current case is false.

3.3 Hidden terminal detection in -HELLO

In the figure[\text{[1]}], \( n_a \), \( n_b \) and \( n_c \) are three different nodes such that the pairs \( (n_a, n_b) \) and \( (n_b, n_c) \) can hear each other but \( n_a \) and \( n_c \) cannot hear each other. Therefore, if \( n_a \) and \( n_c \) simultaneously send messages to \( n_b \), then a signal collision occurs at \( n_b \) resulting into loss of messages which is undesirable. This takes place because \( n_a \) and \( n_c \) are hidden from one another. This is termed as hidden terminal problem.
3.3.1 Hidden terminals are detected using classical protocols

Many active detection mechanism is presented to discover hidden terminals using HELLO messages \[31\]. Whenever a node \( n_i \) wishes to discover hidden terminals, it unicasts a detection request packet to all of its single hop neighbors. Those neighbors of \( n_i \) unicast probe packets to their respective one hop neighbors for a time interval mentioned in the detection request of \( n_i \). If the waiting time of the detection node expires without receiving an ACK, then destination of the corresponding detection probe is assumed to be hidden. In this way, a list of hidden terminals is generated.

Main importance of HELLO message lies in the fact that detection request and detection probe packets are unicast to one hop neighbors and if a node \( n_i \) needs to know about its one hop neighbors, it has to rely on HELLO messages that is broadcast at regular intervals within radio-circle of \( n_i \).

![Fig. 1 Hidden Terminals](image1)

![Fig. 2 Exposed Terminals](image2)

3.3.2 Hidden terminals are detected using -HELLO version protocols

-HELLO versions of protocols follow the similar active mechanism stated in previous subsection with a simple modification. Here detection request and probe packets have to be broadcast within radio-circle of a node so that it reaches all of its 1-hop neighbors. This may require a bit more energy than multiple unicasting of those packets, especially when number of 1-hop neighbors of the node is small. But unicasting detection request and probe packets is not possible without at least one previous broadcast of HELLO message. Therefore, overall cost of hidden terminal detection in classical protocols with more than one HELLO messages between any two consecutive detection requests or probes, is much smaller than the same in protocols based on -HELLO concept. But, if there is exactly one HELLO message between any two consecutive detection requests, the cost of hidden terminal detection with HELLO will be same as protocols based on -HELLO concept.

3.4 Exposed terminal detection in -HELLO

In the figure \[2\] \( n_a, n_b, n_c \) and \( n_d \) are four different nodes such that the pairs \((n_a, n_b), (n_b, n_c), (n_c, n_d)\) can hear each other but the pairs \((n_a, n_c), (n_b, n_d), (n_a, n_d)\) cannot hear each other. Therefore, if \( n_b \) sends a message to \( n_a \) and \( n_c \) simultaneously tries to send a message to \( n_d \), then \( n_c \) will find the medium busy
although these two signals will never collide because they are destined to opposite directions. Therefore \( n_c \) will unnecessarily wait increasing transmission delay in the network.

### 3.4.1 How exposed terminals are detected using classical version protocols

In [32], several methods are described for hidden and exposed terminal detection. If a node keeps itself informed about identification numbers and locations of two hop uplink as well as downlink neighbors, then a lot of exposed terminal problems can be resolved. In this way, \( n_a \) will know about locations of \( n_b \) and \( n_c \); \( n_b \) will know locations of \( n_a \), \( n_c \) and \( n_d \). \( n_c \) will know about positions of \( n_a \), \( n_b \) and \( n_d \) while \( n_d \) will be informed about \( n_b \) and \( n_c \). In this way, if \( n_c \) wants to send a message to \( n_d \) and it knows that \( n_b \) is sending messages to \( n_a \) then \( n_c \) will not delay itself because from location information it will identify that these two signals are in opposite direction and won’t collide with each other. But again knowing about two hop neighbors will require exchanging of HELLO and ACK messages.

### 3.4.2 How exposed terminals are detected using -HELLO version protocols

Detecting exposed terminals will require ACK to RREQs. In -HELLO version of protocols, RREQ message contains information about location and identifiers of all of its one hop uplink neighbors. ACK contains all fields of ACK to HELLO messages along with locations and identification number of its own downlink neighbors. Before sending the first data packet to the specified router in selected path, source broadcasts its 1-hop neighbor information. Therefore, for case in figure [2] when \( n_b \) broadcasts its one hop neighbor information and information about live communication sessions through \( n_b \), then \( n_a \) and \( n_c \) will know about each other. Along with that, \( n_c \) will also be able to identify the directions to which \( n_b \) is going to send messages. Unless transmissions of \( n_c \) are not in same direction, signals generated by \( n_b \) and \( n_c \) won’t collide. In this way, exposed terminal problems can be solved in -HELLO concept.

### 4 SOME -HELLO VERSION PROTOCOLS AS CASE STUDY

In the following subsections, we talk about -HELLO versions of AODV, MM-BCR, MRPC, MTFR, and MFR namely, -HELLO:AODV, -HELLO:MMBCR, -HELLO:MRPC, -HELLO:MTFR, and -HELLO:MFR.

#### 4.1 -HELLO:AODV

##### 4.1.1 Route Discovery

Implementation of -HELLO in reactive routing protocols is convenient because in those protocols, nodes do not have to discover and maintain a route to another node until they need to communicate. In AODV, nodes use HELLO messages for knowing about local connectivity and there exists one hop-count field in the RREQ message. Among the various paths through which a RREQ arrives at the
destination, the one with the minimum hop count is identified for communi-
cation. Minimum hop count value is 1. Maximum possible hop count value depends
upon the total number of nodes in the network. Let it be denoted as $HC$. In -
HELLO:AODV, whenever a node $n_j$ receives a RREQ message from $n_i$, it replies
to $n_i$ using an ACK. After receiving ACK from all downlink neighbors, $n_j$ is ca-
pable of constructing its downlink neighbor table which consists of the attributes:
$<neighbor\_id, neighbor\_location, neighbor\_rad\_range and tmstmp >$.

Here $neighbor\_id$, as the name specifies, is unique identification number of the
neighbor; $neighbor\_location$ is an ordered pair that specifies the last known loca-
tion of the downlink neighbor in terms of latitude and longitude, at timestamp
$tmstmp$. With the help of the downlink neighbor table, each node becomes aware
of the approximate location of the potential successors and can apply transmission
energy optimization during transferring of message packets from one node to an-
other. Hop count field in classical AODV is eliminated in -HELLO:AODV. Each
router appends its own node identifier of the RREQ. When the RREQ will arrive
at the destination, destination node will be able to compute the hop count of the
path. The hop count is ($\alpha + 1$) where $\alpha$ is the number of router-ids appended to
the RREQ.

Here attributes of the RREQ in -HELLO:AODV are:
$<message\_type\_id, source\_id, source\_location, destination\_id, session\_id,\
number\_of\_data\_packets, initiator\_id, maximum\_hop\_count\_difference,\
router\_sequence and timestamp >$.

Here $message\_type\_id$ is 1 for RREQ messages and 3 for RREP messages in -
HELLO:AODV. $source\_id$ and $destination\_id$ specify unique node identifiers of
the source and destination nodes. $session\_id$ is unique identification number of
the communication session between the same pair of source and destination nodes.
The trio $< source\_id, destination\_id, session\_id >$ uniquely identify a RREQ.$initiator\_id$ is equal to $source\_id$ if the RREQ message is intended to begin a new
communication session or when the link between a source node and its immediate
successor has been scrapped and source wants to discover a new route to desti-
nation. On the other hand, $initiator\_id$ will be an identification number of some
router whose link with the corresponding successor (or destination) has been bro-
ken. $maximum\_hop\_count\_difference$ field is set to 0 if $initiator\_id = source\_id$
in the RREQ message, that is, maximum hop count for the current path is
same as maximum possible hop count in the network and hence their difference
is zero; otherwise, $maximum\_hop\_count\_difference$ is ($Z + 1$) where $Z$ is the
number of routers in between the nodes identified by $source\_id$ and $initiator\_id$.

For all RREQ messages intending to repair routes, $message\_type\_id$ will be 2.
$message\_type\_id$ is the field that will differentiate between a fresh RREQ and all
subsequent route-repair efforts by the source. This information is often helpful
for message packet schedulers because route-repair messages are generally given
priority over fresh RREQs. The source is definitely excluded. The field number
of data packets specifies the number of data packets to send from $source\_id$ to
destination $id$ in session $session\_id$.

Knowing the source location is important for the destination because informa-
tion about the optimum route selected by the destination, that needs to come back
to source embed in RREP message. Like classical AODV, whenever a node $n_j$
receives RREQ from $n_i$, it inserts a new entry in the RREQ table where it stores all
attributes of the RREQ message except $message\_type\_id$. Also the corresponding
predecessor-id (i.e. $n_i$) and predecessor location ($x$ and $y$ coordinates of $n_i$) along with current timestamp, are stored in the RREQ table. The predecessor information will be required if the link $n_i \rightarrow n_j$ is present in the optimum path chosen by destination. In that case, $n_j$ shall receive data packets from $n_i$ and send the ACK back to $n_i$. For that, knowing location of $n_i$ will be a prerequisite for $n_j$.

After receiving a RREQ, each node checks whether hop count till that node from the initiator (this can be easily computed from the router’s sequence mentioned in RREQ) is less than or equal to the maximum hop count difference mentioned in the RREQ message or not. If a hop count till that node is really less than or equal to the maximum hop count difference mentioned in the RREQ, then only the node process RREQ further for loop detection; otherwise it is readily discarded. As an example, let us consider Figure 3 where source $n_s$ wants to discover route to a destination $n_d$. Among various paths through which RREQ arrives at the destination, let $n_s \rightarrow n_p \rightarrow n_i \rightarrow n_j \rightarrow n_k \rightarrow n_d$, be a path. Then, $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, 104>$ denote RREQ generated by $n_s$, and $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, null, 100>$, $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, i, 108>$ and $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, i, j, k, 115>$ forwarded by the routers $n_p$, $n_i$ and $n_j$ respectively. We assume as a typical example that $n_s$ generated the RREQ at timestamp 100 that was processed by nodes $n_p, n_i, n_j, n_k, n_x, n_w, n_u$ at timestamps 104, 108, 115, 104, 107 and 110, in that order. Similarly, RREQ packets forwarded by routers $n_x, n_w$ and $n_u$ are: $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, x, 104>$, $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, x, w, 107>$ and $<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, x, w, u, 110>$ respectively.

Assuming that RREQ from $n_s$ arrived at $n_d$ through only the above-mentioned paths, then $n_d$ will choose the route $n_s \rightarrow n_x \rightarrow n_w \rightarrow n_u \rightarrow n_d$, because this is having the hop count 4 whereas the earlier path $n_s \rightarrow n_p \rightarrow n_i \rightarrow n_j \rightarrow n_k \rightarrow n_d$ has hop count 5.

In classical AODV, after processing a RREQ, each router (intending the destination) sets up a reverse path to its predecessor. -HELLO:AODV argues that this is completely unnecessary until and unless the route is really selected for forwarding of data packets. Classical AODV assumes that most links are bidiri-
rectional which is not the case in real life. Therefore, the RREP has to be modeled as another route discovery from destination to source where the sequence of routers in selected optimum route, will be mentioned. Attributes of RREP are: 
<message_type_id, destination_id> (it specifies the node to which a route was intended to be discovered), destination_location, source_id, session_id, initiator_id, maximum_hop_count_difference, current_hop_count, optimum_router_sequence and timestamp >.

current_hop_count field is incremented at each router till HC is not reached. Procedure for loop detection is case of RREP is same as that in case of RREQ. No route maintenance is required for RREP because only after the first data packet is sent through the optimum path, routers will know that they are included in the selected path and therefore, need to set a reverse path to the predecessor, that is, the node from which it received the first data packet. Reverse path setup from a node \( n_i \) to \( n_j \) is easy provided the link is bi-directional. Otherwise, directional flooding [29] is applied to discover a route to \( n_i \). That will not incur much cost because maximum distance between \( n_i \) and \( n_j \) is radio-range of \( n_i \). The format of RREP packets generated by \( n_d \) for \( n_s \) at timestamp 125 is:

\(< 3, d, (X_d(125), Y_d(125)), x, 3, d, 0, 0, x, w, u, 125 >.\)

4.1.2 Loop Detection

After receiving a RREQ a node checks whether hop count till that node is less than or equal to maximum allowable hop count mentioned in that RREQ. If the condition is satisfied, then the receiver of that RREQ consults its RREQ table, to check whether it has received the same RREQ earlier. If one such match is found, then a loop is detected and the newly received RREQ is readily dropped. But if a match is found in that table with only difference in session_id where new session_id is greater than previous session_id, then the new entry replaces previous RREQ entry between the same pair of source and destination nodes. But if new session_id is less than previous session_id between the same pair of source and destination nodes, then it denotes that an unnecessary RREQ has arrived. Hence, it can be readily dropped.

4.1.3 Route Maintenance

In classical AODV, if the source node moves during an active session, it has to re-initiate route discovery procedure to establish a new route to the destination. Similarly, if the destination or some intermediate node moves, a route-break message is sent to the predecessor, because the link from its predecessor to the current router is about to be scrapped. Then the predecessor forwards route-break message to source so that the source can initiate a new route discovery session. Route repair becomes necessary only if more data packets are left to be sent to destination. Periodic HELLO messages are utilized by AODV to detect link failures. On the other hand in -HELLO:AODV, when a node in a live communication path, is about to leave radio-range of its predecessor in that path, it sends a proactive link-fail message (message_type_id for link-fail is 4) to the predecessor. Attributes of this message are: \(< message_type_id, source_id, destination_id, sender_id, predecessor_id and session_id >.\) Here sender_id is the node which is about to get out of radio-circle of its predecessor. Receiving the link-fail message, associated predecessor
sends a repair-request message to source of communication session. Also if the predecessor does not receive ACK of a data packet within a pre-defined time interval, then it sends a repair-request assuming that battery of the successor is exhausted. Attributes of repair-request issued by a router are: 
<message_type_id (5 in case of repair-request) source_id, destination_id, session_id, link_break_timestamp, initiator_id, recv_delay_source >.

All attributes are self-explanatory except link_break_timestamp and recv_delay_source. link_break_timestamp is the timestamp when link breakage was detected by the current router; recv_delay_source specifies the time delay that is required by current node to receive a message from the source. This has been already computed by the current node during transmission of data packets from source. All these information greatly helps in reducing multiple simultaneous repairing efforts by different routers to repair the same route.

Let us consider the situation when in a route R: ns → nx → nw → nu break. Both ns and nw will send a repair-request message to ns as: < 5, s, d, 3, 130, w, βw > and < 6, s, d, 3, w >. Following different cases may occur in that scenario.

Case-1: Both links broke at the same time, ns received repair request from ns. Here, ns accords repair-permission to nx.

Case-2: ns received repair request first from ns and then from nw. Here, ns offers repair permission to nx only since distance between nx and ns is smaller than the same between nw and ns.

Case-3: ns permitted route-repair to nw, then received repair request from ns. In this case, assume that ns received repair requests of nx and nw at timestamps t ns and t w. Therefore, repair permission was given to nw at time t w. Rccv-delay-source of nx and nw are βx and βw, respectively. Therefore, permission granted by nx is supposed to reach nw at time (t w + βw). If t x > (t w + 2βw), then ns expects that nw has received its permission. In that case, ns keeps repair-request of nx. Otherwise, permission is granted to nx too. However, it may happen that, both ns and nw gets repair permission from ns and broadcasts. But that can not cause much harm to the network because < source_id, destination_id, session_id > are same for all those requests and duplicate entries can be easily identified and discarded by routers.

Case-4: None of the routers requesting repair permission could get it from ns. In this case, ns won’t receive route-reply from nd. Maximum waiting timestamp of ns is (t + delRoute + 2 × TTL), where t is the timestamp of generating latest repair permission; delRoute is the time difference between transmitting a data packet from source and getting back its ACK. TTL is time-to-live of a RREQ. (2 × TTL) is the maximum time required for a RREQ packet to reach the destination and fetch corresponding RREP back to the source. After timestamp (t + delRoute + 2 × TTL) × ns, broadcasts RREQ with same < source_id, destination_id, session_id >.

It is expected that distance between the receiver of repair-permission and destination are shorter than the same between source and destination. Therefore, a node that receives a repair permission, broadcasts RREQ to discover a fresh route to the destination, in case more number of packets are to be sent. Due to a comparatively close position with respect to the destination, number of RREQs produced by the said receiver of repair permission, is generally much lesser than the same produced by the source. This is shown in figure 5. Link from nw to nu is about to be broken because nu will get out of the radio-circle of nw very soon. So, nu
sends a link-fail message to $n_w$ and $n_w$ as: $<4, s, d, u, w, 3>$. After receiving the link-fail message, $n_w$ will broadcast RREQ as: $<5, s, d, 3, 130, w, \beta_w>$ to discover a new route to $n_d$. From figure 4 it can be clearly seen that distance between $n_w$ to $n_d$ is much smaller than distance between $n_s$ and $n_d$. Therefore, if $n_w$ initiate directional route discovery (directional flooding, as recent location of the destination is known), then the cost of RREQ packets will be much lesser than if $n_s$ initiates route discovery. If we assume that the link from $n_w$ to $n_u$ broke at timestamp 130 after sending three data packets to the destination and $n_w$ obtained permission for the route-repair from $n_s$ at time 137. Then RREQ generated by $n_w$ will look like $<5, s, d, 3, 130, w, \beta_w>$. Here number of packets to be transmitted is 2.

### 4.1.4 Transmission Power Optimization

In AODV, power optimization is performed based on location information of downlink neighbors when they acknowledge HELLO messages of their uplink neighbors. But in -HELLO:AODV, this is not possible because HELLO messages are not periodically sent. Therefore, to implement transmission power optimization in -HELLO:AODV, proactive ACK are sent from a node to some of its successors; those successors have to be connected to that node through live communication sessions. Interval between two consecutive proactive ACK is same as the one between two consecutive HELLO messages. Transmission of proactive ACK from $n_j$ to $n_i$ will continue till all communication sessions utilizing link from $n_i$ to $n_j$, complete. Components of this proactive ACK are similar to the HELLO message mentioned in section 3, with only one additional field, namely, minimum-receive-power; the name says it all. Proactive ACK gives $n_i$ information about the most recent location of $n_j$. This is used by $n_i$ while it sends next data packet to $n_j$, whatever live session it may be. Let, location of $n_j$ at time $t$ be $(x_j(t), y_j(t))$ where $t$ is timestamp of last proactive ACK from $n_j$ to $n_i$. If minimum received power of $n_j$ be denoted as $\text{minRecv}(j)$, then minimum required transmission power $\text{transPower}_i(j,t)$ required by $n_i$ to send a packet to $n_j$ at time $t$, is formulated in (21). This formulation is as per Frii’s transmission equation [33, 9].

$$\text{transPower}_i(j,t) = \text{minRecv}(j) \times \text{dist}^2_{ij}(t)/C$$  \hspace{1cm} (21)

$$\text{dist}_{ij}(t) = \sqrt{(X_j(t)-X_i(t))^2 + (Y_j(t)-Y_i(t))^2}$$  \hspace{1cm} (22)
Transmission power required by \( n_i \) to send a data packet to \( n_j \) without power optimization, is denoted as \( \text{transNonOpt}_i \) and defined in equation (23).

\[
\text{transNonOpt}_i = \min\text{Recv}(j) \times R_i^2 / C \tag{23}
\]

where \( R_i \) is radio-range of \( n_i \). Therefore, transmission power \( \text{savedPower}_i(j,t) \) saved in \( n_i \) after optimization based on proactive ACK of \( n_j \), at time \( t \), is given by equation (24).

\[
\text{savedPower}_i(j,t) = \text{transNonOpt}_i - \text{transPower}(j,t) \tag{24}
\]

### 4.1.5 Comparing sizes of various messages in AODV and -HELLO:AODV

During comparison of message sizes in AODV and -HELLO:AODV, first comes the RREQ. Additional attributes in RREQ packet of -HELLO:AODV are: \(<\text{initiator id}, \text{maximum hop count difference}, \text{router sequence}>\). In classical AODV, router sequence was not there because AODV assumed that links are all bi-directional and therefore, maintaining a link to the immediate previous node was sufficient. But in general, link are not bi-directional. So, it is not sufficient to keep track of immediate predecessor because RREP can not always be sent in the reverse link. Hence, in a network consisting of mostly uni-directional links, router-sequence information needs to be maintained. Eliminating router-sequence as an additional RREQ attribute, we are left with \( \text{initiator id} \) and \( \text{maximum hop count difference} \). If \( N \) denotes the total number of nodes in the network, number of bits required to represent \( \text{initiator id} \) is \( \log_2 N \). Maximum value of hop count difference is \( H \) where \( H \) is maximum possible hop count in the network \([34]\).

Theorem 1 proves that, \( \log_2 H = \log_2 X + \log_2 Y + \log_2 \sqrt{X^2 + Y^2} - 3 - \log_2 N - \log_2 R_{min} \). So, total number of bits \( B_{add\_RREQ} \) required to represent the additional attributes of RREQ in -HELLO:AODV, is defined in equation (25).

\[
B_{add\_RREQ} = \log_2 H + \log_2 N \\
= \log_2 X + \log_2 Y + \log_2 \sqrt{X^2 + Y^2} - 3 - \log_2 R_{min} \tag{25}
\]

Since, \( XY > \sqrt{X^2 + Y^2} \) for \( X, Y > 2 \), so, \( B_{add\_RREQ} \) \( < 2(\log_2 X + \log_2 Y) \). \( R_{min} \) is minimum radio-range among all nodes in the network. Number of bits required to represent each attribute of HELLO message (it is applicable to classical versions of all the protocols in MANETs), is shown as below:

1. message type id (3 bits)
2. sender id (\( \log_2 N \) bits)
3. sender location (\( \log_2 X + \log_2 Y \) bits)
4. radio range (\( \log_2 R_{max} \) bits)
5. current time stamp (\( \log_2 TM \) bits)

\( TM \) is the total simulation time and \( R_{max} \) is maximum radio-range among all nodes in the network. So, total number of bits \( B_{HELLO} \) required to represent a HELLO message, appears in equation (26).

\[
B_{HELLO} = 3 + \log_2 N + \log_2 X + \log_2 Y + \log_2 R_{max} + \log_2 TM \tag{26}
\]

From equations (25) and (26) we get, \( B_{add\_RREQ} < 2 \times B_{HELLO} \)
It is clear from the above inequality that number of bits required to represent additional attributes in RREQ message is less than two HELLO messages.

**Theorem 1:** \( \log_2 H = \log_2 X + \log_2 Y + \log_2 \sqrt{X^2 + Y^2} - 3 - \log_2 N - \log_2 R_{min} \)

**Proof:** Let \( P \) and \( D \) denote average progress in each hop from source to destination and average distance between a source and destination. Therefore \( H \) can be estimated as \( H = D/P \). Average one hop progress \( P \) is approximated as the maximum distance between a sender and each of the neighbors within its transmission range. Average number of nodes in the circle of radius \( R_{min} \), is denoted as \( \xi \) and defined in equation (27).

\[
\xi = \frac{(N/(XY))\pi R_{min}^2}{27}
\]

The probability of all \( \xi \) nodes residing within distance \( r \) from center of transmission circle can be formulated as in equation (28).

\[
F(r) = \text{Prob}(\text{all } \xi \text{ nodes residing within distance } r) = [\text{Prob}(\text{a node reside within } r)]^{\xi} = [\pi r^2 / \pi R_{min}^2]^{\xi} = r^{2\xi} / R_{min}^{2\xi}
\]

we have assumed independence and randomness node location. The probability density function (pdf) of progress \( r \) from source, is given by equation (29).

\[
f(r) = \frac{\partial F(r)}{\partial r} = 2\xi r^{2\xi-1} / R_{min}^{2\xi}
\]

Therefore, average progress is then the expected value of \( r \) with respect to pdf \( f(r) \), can be calculated as in equation (30).

\[
I = \int_0^{R_{min}} r f(r) dr = 2\xi R_{min} / (2\xi + 1)
\]

In a network of size \((X \times Y)\), average distance \( D \) between source and destination, is approximated as:

\( D \approx (0 + \sqrt{X^2 + Y^2})/2 \)

Therefore expected number of hops\( (H) \) is,

\[
H \approx D/P \approx \sqrt{X^2 + Y^2} / 2I \approx (2\xi + 1) \sqrt{X^2 + Y^2} / 4\xi R_{min}
\]

Therefore,

\[
\log_2 H = - \log_2 2 + \log_2 X + \log_2 Y + \log_2 \sqrt{X^2 + Y^2} - 3 - \log_2 N - \log_2 R_{min}
\]

Hence by equation (32) proves the theorem.

As far as the link-fail message is concerned, it is completely new in -HELLO:AODV. Number of bits required to represent a link-fail message, is computed as follow:

1. message type id(3 bits)
2. source id(log\_2 N bits)
3. destination id (log$_2 N$ bits)
4. link breakage detector id (log$_2 N$ bits)
5. link broke with node id (log$_2 N$ bits)
6. remaining number of packets (log$_2 PAC$ bits)

Here, $PAC$ is upper limit of total number of packets that can be transmitted in a session from any source to the destination. Theorem 2 proves that link-fail does not impose any additional byte overhead.

**Theorem 2:** Link Fail does not require any additional byte.

**Proof:** If link-fail message is not sent, then data packet has to be sent thrice, that is, two times more than the number of times a data packet is sent in the classical case. Format of a data packet is expressed as follow:

1. message type id(2 bits)
2. source id (log$_2 N$ bits)
3. destination id (log$_2 N$ bits)
4. session id (log$_2 TM$ bits)
5. packet sequence id (log$_2 PAC$ bits)

Therefore, additional number of bits required to represent a link-fail message compared to two ordinary data packets, is denoted as $B_{add\_Link\_fail}$ and defined in (33).

$$B_{add\_Link\_fail} = 3 + 4 \log_2 N + \log_2 PAC$$
$$-2(3 + 2 \log_2 N + \log_2 TM + \log_2 PAC)$$
$$= -3 - \log_2 PAC - 2 \log_2 TM$$

(33)

Therefore, $B_{add\_Link\_fail} < 0$

Hence, this proves, link-fail does not require any additional byte. So, this is an improvement produced by -HELLO:AODV over classical AODV.

Repair-request message also exists in classical AODV. The node that discovers link breakage sends a message to the source informing link breakage so that source can initiate route repair. As far as repair-permission message is concerned, it is additional in -HELLO:AODV. Theorem 3 specifies that the additional byte requirement imposed by repair-permission is covered by four HELLO messages.

**Theorem 3:** $B_{repair\_permission} < (4 * B_{HELLO})$.

**Proof:** Number of bits required by a repair-permission is shown as follow:

1. message type id(3 bits)
2. source id (log$_2 N$ bits)
3. destination id (log$_2 N$ bits)
4. packet sequence id (log$_2 PAC$ bits)
5. link breakage detector id (log$_2 N$ bits)

So, total number of bits required by a repair-permission is denoted as $B_{repair\_permission}$ and formulated in (34).

$$B_{repair\_permission} = 3 + 3 \log_2 N + \log_2 PAC$$

If we assume that $PAC < N$, then,

$$B_{repair\_permission} \approx 3 + 3 \log_2 N$$

(34)
So, \( B_{\text{repair permission}} < (4 * B_{\text{HELLO}}) \)

If \( x \) number of HELLO messages are saved, then saved energy \( SE \) is given by:

\[
SE = x * \text{avg}\_eng\_HELLO(i)
\]

4.2 -HELLO:MMBCR

4.2.1 Route Discovery

In -HELLO embedded version of MMBCR, the source node appends its residual energy information with RREQ message. This field is called minimum-residual-energy. After it is received by the first router, it checks whether its own residual energy is less than that embedded within the RREQ message. If the condition is satisfied, then the router replaces minimum residual energy in RREQ message with its own residual energy which becomes new minimum-residual-energy of the RREQ packet that will be forwarded by the current router. On the other hand, if the residual energy of the current router is greater than or equal to the minimum-residual-energy mentioned in the RREQ it has received, then the current router does not change minimum-residual-energy of the RREQ packet while forwarding it. Except minimum-residual-energy, all other fields of the RREQ are similar to -HELLO:AODV. For example, RREQ generated by \( n_s \) is as: \(< 1, s, (X_s(1), Y_s(1)), d, 3, 5, s, e_j(9), r_j(9), f(s, j), s, i, j, 0, 0 >\) and network scenario shown in figure 3 will look like: \(< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, \text{null}, 100, 4 >\) where we have assumed that residual energy of \( n_s \) is 4 J. Also assume that residual energy of \( n_p \) at timestamp 104 is 2 J as in: \(< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, 104, 2 >\) and the same of \( n_i \) at timestamp 108 is 5 J shown here: \(< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, i, 108, 2 >\).

Here \( n_p \) changed minimum-residual-energy of RREQ sent by \( n_s \) from 4 J to 2 J because the minimum residual energy of \( n_p \) is 2 J which is less than 4 J. But \( n_i \) did not change minimum-residual-energy of the RREQ it received from \( n_p \) because the residual energy of \( n_i \) is 5 J which is higher than minimum-residual-energy (2 J) embedded in RREQ sent by \( n_p \) to \( n_i \). RREQs arrive at the destination through multiple paths. All these paths have a minimum-residual-energy. Among them, the path with maximum of these minimum-residual-energies, is selected for communication.

Loop Detection, Route Maintenance and various message sizes of -HELLO:MMBCR are same as -HELLO:AODV.

4.3 -HELLO:MRPC

4.3.1 Route Discovery

In -HELLO version of MRPC, source node \( n_s \) includes an information \( f\_Eng(s) \) with the RREQ packet where \( f\_Eng(s) \) is defined in equation (35).

\[
f\_Eng(s) = \frac{\text{resEng}(s)}{\text{unitPktEng}(s)} \quad (35)
\]

Here, \( \text{resEng}(s) \) and \( \text{unitPktEng}(s) \) denote current residual energy of \( n_s \) and energy required by \( n_s \) to transmit one packet, respectively. After the first router \( n_p \)
receives that RREQ, it checks whether $f_{Eng}(p) < f_{Eng}(s)$ or not. If as, then $f_{Eng}(s)$ is replaced by $f_{Eng}(p)$ in the RREQ packet before it is forwarded to the next router. Next router follows a similar procedure. All other fields of the RREQ are same as -HELLO:AODV. For example, RREQ generated by $n_s$ in context of $< 1, s, (X_s(1), Y_s(1)), d, 3, 5, s, e_j(9), r_j, v_j(9), f(s, j), s, i, j, 9, 0 >$ and network scenario shown in figure 3 will look like: $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, null, 100, 1000 >$. Where we have assumed that residual energy of $n_s$ is 4J. Also assume that residual energy of $n_p$ at timestamp 104 is 2J and the same of $n_i$ at timestamp 108 is 5J. The energy required for transmission of one packet is 4mJ for $n_s$, 20mJ for $n_p$ and 10mJ for $n_i$. RREQs forwarded by $n_p$ and $n_i$ are as: $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, 104, 100 >$ and $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, i, 108, 100 >$. Residual packet capacity of $n_s$ is $f_{Eng}(s)$ which evaluates to (4J/4mJ) i.e. 100. The same of $n_p$ is (2J/20mJ) i.e. 100 which is less than the residual packet capacity of $n_s$. Therefore $n_p$ updates the last field of RREQ from 1000 to 100. But $n_i$ did not change it because $f_{Eng}(i)$ is (5J/10mJ) i.e. 500. RREQs arrive at the destination through multiple paths. All these paths have a residual packet capacity. Among them, the path with maximum $f_{Eng}$ is selected for communication.

Loop Detection, Route Maintenance and various message sizes of -HELLO:MRPC are same as -HELLO:AODV.

4.4 -HELLO:MTPR

4.4.1 Route Discovery

In -HELLO:MTPR, source node includes a special transmission power field with RREQ packet which is initially set to null. After the first router $n_p$ receives RREQ packet, it computes minimum transmission power required by $n_s$ to send a message to $n_p$, as in equation (1). This is possible for $n_p$ because $n_p$ knows its own location and minimum receive power requirements. Location of $n_s$, too, is known to $n_p$ from RREQ received from $n_s$. Considering the context of MMBCR, RREQ generated by $n_s$ forwarded by $n_p$ and $n_i$ are as:

- $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, null, 100, null >$,
- $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, 104, 100 >$ and
- $< 1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, i, 108, 100, F(s, p, i) >$.

Where $F(s, p, i) = min(transPower_s(p, 104), transPower_p(i, 108))$. RREQs arrive at the destination through multiple paths. All these paths have a minimum transmission power. Among them, the path with minimum of these minimum transmission powers, is selected for communication.

Loop Detection, Route Maintenance and various message sizes of -HELLO:MRPC are same as -HELLO:AODV.
4.5 -HELLO:MFR

4.5.1 Route Discovery

In -HELLO:MFR, source node and each router append their node identifier along with current location so that destination can decide the optimum route based on comparative distances of downlink neighbors of a router, from the router itself. This is projected on the line connecting source and destination of a communication session. Considering the context of MFR, RREQ generated by \( n_s \) and forwarded by \( n_p \) and \( n_i \) are as:

\[
<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, 100 >,
<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, (X_p(104), Y_p(104)), 104 > \text{ and }
<1, s, (X_s(100), Y_s(100)), d, 3, 5, s, 0, p, (X_p(104), Y_p(104)), i, (X_i(108), Y_i(108)), 108 >.
\]

Other fields are as per AODV counterpart.

4.6 -HELLO:ABR: Our opinion

Based on the idea of encapsulating information in HELLO messages within RREQs, other MANET routing protocols that do not directly depend on HELLO messages (FORP, DSR etc.) are also convertible to their -HELLO versions, except associativity based routing or ABR. As per our study, ABR is a routing protocol that directly depends upon HELLO messages and without using HELLO and ACK, it is almost impossible to find out at what time a node enters radio-range of some other node. Stability in ABR is measured by the number of associativity ticks received by a node from some of its predecessors. Associativity ticks are nothing but periodic HELLO messages. Therefore, HELLO messages cannot be omitted in ABR. Still, for energy optimization, we can eliminate periodic ACK messages (ACK without any live communication). In that case, downlink neighbor of a node will still be aware of the number of associativity ticks it has been receiving continuously and after a RREQ arrives from a predecessor, it will be able to evaluate stability of the link from its predecessor to the current node and include that within RREQ before forwarding it to the next router. In this way, the algorithm can function.

5 SIMULATION

5.1 Simulation Environment

In simulation experiments, performance analysis of these algorithms is done using network simulation (NS-2) version 2.33. Simulation parameters appear in table 1. -HELLO versions of the protocols AODV, MMBCR, MTPR, MRPC and MFR are compared with their classical versions. Simulation metrics are energy consumption (in mJ), network lifetime (in seconds), average end-to-end delay per session and network throughput (percentage of data packets that could reach their respective destinations).
Table 1 Simulation Parameters

| Parameter                                      | Specification                           |
|------------------------------------------------|-----------------------------------------|
| Topology Area                                  | 500 x 500 (square meter)               |
| Traffic type                                   | Constant bit rate (CBR)                |
| Packet size                                    | 512 bytes                               |
| HELLO packet interval in classical versions    | 10 milliseconds                         |
| Node mobility                                  | 10-30 (meter per seconds)              |
| Signal frequency                               | 2.4 GHz                                 |
| Channel capacity                               | 2 Mbps                                  |
| Transmission power                            | 300-600 mW                              |
| Receiving power                                | 50-300 mW                               |
| Mobility model                                 | Random waypoint                         |
| Radio range                                    | 50 to 100 meter                         |
| Initial energy of nodes                        | 5 J to 10 J                             |
| Pause time                                     | 1 second                                |
| Number of nodes                                | 20, 40, 60, 70, 80, 90, 100             |

Fig. 5 Graphical illustration of energy consumption vs number of nodes

5.2 Simulation Results

Simulation graphs appear in the figures 5 to 20. Explanation these graphs given below with reference to performance metrics namely, energy consumption, network lifetime, end-to-end delay and network throughput.

5.3 Energy Consumption

Compared to classical versions of the protocols (AODV, MMBCR, MRPC, MTPR and MFR) with -HELLO versions, energy consumption in nodes are greatly reduced. It has been shown in this article that by simple alteration of structures in RREQ message, HELLO messages can be avoided especially in reactive, energy-aware and stability oriented routing protocols. Moreover, route maintenance in -HELLO embedded protocols, is performed in such a manner that it consumes less energy than route maintenance in classical versions of those protocols. Whenever a link breakage is detected by a router in a live communication path, classical protocols instruct the router to send that information to the associated source
of communication, so that the source can re-initiate a route discovery process. -HELLO version protocols emphasize on the fact that distance of current router, that has discovered link breakage, from the destination, is expected to be significantly smaller than the same between source and destination. This leaves a deep impact on energy consumption in route re-discovery. If source initiates route re-discovery, then more RREQ packets will be generated compared to the situation when route rediscovery is initiated by a router than has discovered link breakage. Injection of more RREQ packets means that those packets have to be forwarded by other nodes in the network, increasing energy consumption by nodes. This is seen by figures 5, 9, 13 and 17. As expected, energy consumption increases with number of nodes and also with increase in packet load (as per figures 9 and 17).

However, energy consumption in AODV is much higher than others because AODV is not concerned with energy of nodes. It selects the path with minimum hop count, as optimal. MFR, although does not directly associate its optimum path selection criteria with residual energies of nodes, but still it tries to minimize pair-wise distance between consecutive routers. So, transmission power required by source and each router is minimum possible in case of MFR. MRPC, MTPR and MMBCR are already energy aware but still eliminating HELLO ensures great im-
Fig. 8 Graphical illustration of network throughput vs number of nodes

Fig. 9 Graphical illustration of energy consumption vs packet load

Fig. 10 Graphical illustration of network lifetime vs packet load
Fig. 11 Graphical illustration of end-to-end delay per session vs packet load

Fig. 12 Graphical illustration of network throughput vs packet load

Fig. 13 Graphical illustration of energy consumption vs number of nodes
Fig. 14 Graphical illustration of network lifetime vs number of nodes

Fig. 15 Graphical illustration of end-to-end delay per session vs number of nodes

Fig. 16 Graphical illustration of network throughput vs number of nodes
Fig. 17 Graphical illustration of energy consumption vs packet load

Fig. 18 Graphical illustration of network lifetime vs packet load

Fig. 19 Graphical illustration of network lifetime vs packet load
Fig. 20 Graphical illustration of network throughput vs packet load

provement. This improvement is 50.17% for AODV, 41.67% for MMBCR, 42.46% for MRPC, 40.48% for MTPR, 45.25% for MFR.

5.4 Network Lifetime

With increase in energy consumption, lifetime of nodes decreases. If a node participating in a live communication dies, then link breakage will be detected by its predecessor and in order to repair the broken link, more RREQ packets are injected into the network. That consumes more energy in nodes resulting in death of more nodes. This is an ominous circle. In schemes lifetime improvements are: 48.48% for AODV, 39.22% for MMBCR, 35.56% for MRPC, 24.57% for MTPR, 29.64% for MFR. As seen from the figures 6, 10,14 and 18, network lifetime increases with increase in number of nodes and when number of nodes is fixed and packet load varies, network lifetime reduces.

5.5 End-to-end Delay

Phenomena like route re-discoveries increase end-to-end delay in a communication session. Reason is that transferring data packets can not start until and unless link breakage is repaired. As mentioned earlier, repairing of link breakage means broadcasting huge number of RREQs and it is a time consuming process. Time duration required for route re-discovery increases end-to-end delay in a communication session. In this scheme delay improvement are: 32.52% for AODV, 43.45% for MMBCR, 25.96% for MRPC, 37.75% for MTPR, 22.97% for MFR. The figures 7, 11,15 and 19 are refering in this respect.

5.6 Network Throughput

Network throughput is greatly influenced by route re-discovery. An increased amount of injected RREQ packet cause greater number of contention and packet
collision in the network. Also network lifetime is reduced and as a result fewer packets can successfully reach their respective destinations. Here throughput improvement are: 11.09% for AODV, 9.33% for MMBCR, 8.96% for MRPC, 9.75% for MTPR, 8.27% for MFR. For all the protocols it is seen that initially network throughput increase with increases in number of nodes and later it decreases. Initial improvement is due to better network connectivity whereas after the network becomes dense or highly populated, network throughput starts decreasing. As expected, network throughput decreases with increase in packet load. These findings are evident from the figures 8, 12, 16 and 20.

6 Conclusion

From the perspective of mobile ad-hoc communications, it is extremely important to save battery power as much as possible. That will lead to increase the lifetime of nodes ensuring thereby prolonged opportunity to forward packets (both data and control) of others. This HELLO version protocols reduce energy consumption by reducing or eliminating RREQ packet that is HELLO message. It also reduces link breakage phenomenon as most of the link breakage occurs due to exhausted batteries power of nodes. Therefore, the number of RREQ packets injected into the network for repairing of those routes, also reduce. This also leads to decrease end-to-end delay, increase throughput. The simulation analysis showing good performance improvements. Extensive simulation analysis and testing in real testbed environment could be one the future scope of the scheme.

References

1. I. Chlamtac, M. Conti, and J. Liu, “Mobile ad hoc networking: imperatives and challenges,” Ad Hoc Networks, vol. 1, pp. 13 – 64, July 2003.
2. M. Conti and S. Giordano, “Mobile ad hoc networking: milestones, challenges, and new research directions,” IEEE Communications Magazine, vol. 52, no. 1, pp. 85–96, 2014.
3. D. Helen and D. Arivazhagan, “Applications, advantages and challenges of ad hoc networks,” Journal of Academia and Industrial Research (JAIR), vol. 2, pp. 453 – 457, 2014.
4. J. Vazifehdan, R. V. Prasad, and I. Niemegeers, “Energy-efficient reliable routing considering residual energy in wireless ad hoc networks,” IEEE Transactions on Mobile Computing, vol. 13, no. 2, pp. 434–447, 2014.
5. A. Sufian, A. Banerjee, and P. Dutta, “Energy and velocity based tree multicast routing in mobile ad-hoc networks,” Wireless Personal Communications, vol. 107, pp. 2191–2209, Aug 2019.
6. S. Corson and J. Macker, “Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations,” https://tools.ietf.org/html/rfc2501.html, January 1999.
7. A. Banerjee, A. Sufian, and P. Dutta, “Emr-pl: Energy-efficient multipath routing based on link life prediction in ad hoc networks,” Journal of Information and Optimization Sciences, vol. 39, pp. 285–301, Sept 2017.
8. S. Singh, M. Woo, and C. S. Raghavendra, “Power-aware routing in mobile ad hoc networks,” in Proceeding of the 4th annual ACM/IEEE international conference on Mobile computing and networking(MobiCom ’98), pp. 181 – 190, 1998.
9. M. Abolhasan, T. Wysocki, and E.Dutkiewicz, “A review of routing protocols for mobile ad hoc networks,” Ad Hoc Networks, vol. 2, pp. 1–22, 2004.
10. C. Perkins and P. Bhagwat, “Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers,” ACM Computer Communication Review, vol. 24, pp. 234–244, Oct 1994.
11. T. Chen and M. Gerla, “Global state routing: a new routing scheme for ad-hoc wireless networks,” in *Proceeding of IEEE International Conference on Communications*, 1998, June 1998.

12. S. Murthy and J. J. Garcia-Luna-Aceves, “An efficient routing protocols for wireless networks,” *ACM Mobile Networks and Application Journal*, pp. 183–197, Oct 1996.

13. G. Pei, G. Gerla, and T. Chen, “Highly dynamic destination-sequenced distance-vector routing (dstv) for mobile computers,” *ACM Computer Communication Review*, vol. 24, pp. 71–78, Oct 1994.

14. C. Perkins and E. Royer, “Ad hoc on demand distance vector routing, mobile computing systems and applications,” in *Proceeding of WMCSA 99*, pp. 90–100, 1999.

15. D. Johnson, D. Maltz, and J. Broch, “Dsr: the dynamic source routing protocol for multi-hop wireless ad hoc networks,” in *Ad Hoc Networking* (C. Perkins, ed.), ch. 5, pp. 139–172, Addison-Wesley, 2001.

16. W. Su and M. Gerla, “Ipv6 flow handoff in ad hoc wireless networks using mobility prediction,” in *Proceeding of IEEE GlobeCom*, 1999.

17. V. Park and S. Corson, “Temporally-ordered routing algorithm (tora) functional specification,” *IETE Internet draft*, 1997.

18. Y. Ko and N. H. Vaidya, “Location aid routing (lar) in mobile ad hoc networks,” in *Proceeding of ACM/IEEE Mobicom*, 1998.

19. T. Hou and V. Li, “Transmission range control in multi-hop packet radio networks,” *IEEE Transactions on Communications*, vol. 34, Jan 1986.

20. A. Misra and S. Banerjee, “Mrpc: maximizing network lifetime for reliable routing in wireless environments,” in *Proceeding of WCNC*, 2002.

21. C. K. Toh, “Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks,” *IEEE Communications Magazine*, pp. 138–145, June 2001.

22. N. Meghanathan, “Survey and taxonomy of unicast routing protocols for mobile ad hoc networks,” *International journal of applications of graph theory in wireless ad hoc and sensor networks*, vol. 1, Dec 2009.

23. A. Aref and S. Ulukus, “Mobile energy harvesting nodes: Offline and online optimal policies,” *IEEE Transaction on Green Communications and Networking*, vol. 2, pp. 367–382, March 2018.

24. A. Banerjee, P. Dutta, and A. Sufian, “Movement guided management of topology (msgmt) with balanced load in mobile ad hoc networks,” *Int. j. info. technol.*, vol. 11, pp. 149–158, 2019.

25. A. Banerjee, P. Dutta, and A. Sufian, “Fuzzy-controlled energy-efficient single hop clustering scheme with (fesc) in ad hoc networks,” *Int. j. info. technol.*, vol. 10, pp. 313–327, 2018.

26. M. G. vaighan and M. A. J. Jamali, “A multipath qos multicast routing protocol based on link stability and route reliability in mobile ad-hoc networks,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, pp. 107–123, Jan 2019.

27. C. K. Toh, “Associativity based routing for ad hoc mobile networks,” *Wireless Personal Communications*, vol. 4, pp. 103–139, March 1997.

28. S. Mohseni, R. Hassam, A. Patel, and R. Razali, “Comparative review study of reactive and proactive routing protocols in manets,” in *4th IEEE International Conference on Digital Ecosystems and Technologies*, pp. 304–309, April 2010.

29. A. Banerjee and P. Dutta, “Fuzzy controlled adaptive and intelligent route (fair) selection in mobile ad hoc networks,” *European Journal of Scientific Research*, vol. 45, pp. 367–382, Sept 2010.

30. S. Gupta, M. Yadav, and R. Saket, “Mathematical analysis for stability based routing in ad hoc networks,” *Prayna Research Journal*, vol. 60, 2014.

31. F. Y. Li, A. Kristensen, and P. Engelstad, “Hidden terminal detection in 802.11-based wireless ad hoc networks,” in *Proceedings of the 15th IST Mobile and Wireless Communication Summit*, 2006.

32. Z. J. Haas and J. Deng, “Dual busy tone multiple access (dbtma)-a multiple access control scheme for ad hoc networks,” *IEEE Transactions on Communications*, vol. 50, no. 6, pp. 975–985, 2002.

33. I. Kaur and A. L. N. Rao, “A framework to improve network security with less mobility in manet,” *International Journal of Computer Applications*, vol. 167, pp. 21–24, June 2017.

34. K. Liu, J. Deng, P. Varshney, and R. Balakrishnan, “An acknowledgement-based approach for detecting routing misbehavior in manets,” *IEEE Transaction on Mobile Computing*, vol. 6, pp. 536–550, June 2007.