An Energy-Efficient Broadcast Protocol in MANETs: Design and Evaluation

Name: Xiaoying Zhang

Date: Jan. 12, 2010

SID: 100782755

ABSTRACT

As the technology of mobile ad hoc networks (MANETs) develops, many new kinds of applications in this field emerge. The group-oriented services which take advantage of the broadcasting nature of wireless networks are of much importance. Therefore, broadcasting/multicasting protocols in MANETs are receiving increased attention. Energy efficiency is a critical issue in MANETs and sensor networks where power of nodes is limited and difficult to recharge. This issue is crucial in the design of new routing protocols since each node acts not only as a host but also as a router. This project gives a general survey of broadcast/multicast routing protocols, network coding approaches and energy-efficient broadcast/multicast routing protocols in MANETs. In order to maximize network lifetime, we propose a new energy-efficient broadcast protocol, called EBOLSR, which adapts the EOLSR protocol to the broadcasting domain. And then we compare the performance of EBOLSR with three other broadcast protocols in two distinct MANET scenarios, Classical Flooding, Simplified Multicast Forwarding (SMF), and a coding-based broadcast protocol (CodeBCast). Simulation results show that EBOLSR protocol has less energy consumption and longer network lifetime than Classical Flooding, and also explain the reason why it does not outperform SMF in terms of the energy consumption and network lifetime.

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my supervisor, Professor Thomas Kunz, for his guidance and support during my research and study at Carleton University. Without his guidance, patience and persistent help, this project would not have been possible.

I also would like to heartily thank my co-supervisor, Professor Oliver Yang, whose encouragement, guidance and support helped me a lot when I was studying in the University of Ottawa. His perpetual energy and enthusiasm in research had motivated me.

In addition, I would like to give my thanks to my parents. Their endless love and considerations help me to overcome all the difficulties confidently. I cannot ask for more from them. And I only want to let them know they will be proud of their daughter.

Lastly, I offer my regards and blessings to all of those who have helped me in any respect during my life in Canada.

Contents

1 INTRODUCTION - 4 -
2 MULTICAST/BROADCAST PROTOCOLS - 8 -
2.1 Multicast Protocols - 8 -
2.1.1 Protocols Classification - 8 -
2.1.2 Simplified Multicast Forwarding (SMF) - 10 -
2.2 Broadcast Methods - 10 -
2.2.1 Classical Flooding - 11 -
2.2.2 Probability-Based Methods - 11 -
2.2.3 Area-Based Methods - 12 -
2.2.4 Neighbor Knowledge Methods - 13 -
3 ENERGY-EFFICIENT MULTICAST/BROADCAST PROTOCOLS - 15 -
3.1 MEB/MEM - 15 -
3.1.1 MEB/MEM with Omni-directional Antennas - 16 -
3.1.2 Algorithm Requirement and Latest Development - 17 -
3.2 MLB/MLM - 19 -
3.2.1 MLB/MLM with Omni-directional Antennas - 19 -
3.2.2 Algorithm Requirement and Latest Development - 20 -
4 NETWORK CODING - 22 -
4.1 Description of Network Coding - 22 -
4.2 Network Coding in MANETs - 24 -
4.2.1 Subgraph Selection - 24 -
4.2.2 Wireless Medium Contention - 25 -
4.2.3 Coding Efficiency - 26 -
4.2.4 Complexity of Encoding and Decoding - 27 -
4.2.5 Energy Efficiency - 27 -
5 ENERGY-EFFICIENT BROADCAST OLSR (EBOLSR) and Other Algorithms for Comparison - 28 -
5.1 EBOLSR - 28 -
5.1.1 Motivation - 28 -
5.1.2 Energy Consumption Model - 29 -
5.1.3 Hello Message Format - 29 -
5.1.4 Energy-Efficient Selection of MPRs - 30 -
5.2 Classical Flooding - 32 -
5.3 SMF - 33 -
5.4 CodeBCast - 34 -
6 PERFORMANCE EVALUATION - 37 -
6.1 Network Scenarios and Performance Metrics - 37 -
6.1.1 WiFi Scenario - 37 -
6.1.2 RiceanRescue Scenario - 38 -
6.1.3 Performance Metrics - 40 -
6.2 Performance Comparison - 41 -
6.2.1 WiFi Scenario - 41 -
6.2.2 RiceanRescue Scenario - 45 -
7 CONCLUSIONS - 48 -

1 INTRODUCTION

A mobile ad hoc network (MANET) is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration. Due to the limited transmission range of wireless network interfaces, to communicate with nodes outside its transmission range, a node needs multiple hops to forward packets to the destination across the network. Since there is no stationary infrastructure such as base stations, each node operates not only as a host but also as a router. Hence, a routing protocol for MANETs runs on every node and is affected by the resources at each mobile node. Considering typical characteristics of a MANET, such as a lack of infrastructure, dynamic topologies, constrained bandwidth, constrained energy and so on, a good routing protocol should minimize the limited resources and meanwhile maximize the network efficiency.

In recent years, a variety of routing protocols have been proposed for MANETs. Such protocols can be classified as proactive or reactive, depending on whether they keep routes continuously updated, or whether they react on demand. They can also be classified as unicast routing, broadcast routing and multicast routing, according to the type of applications. Unicast routing supports communications between one source and one destination. Dynamic Source Routing (DSR) [1], Ad Hoc On-demand Distance Vector (AODV) [2], Destination Sequenced Distance Vector (DSDV) [3], Optimized Link State Routing (OLSR) [4] protocol and so on are the typical unicast routing protocols proposed for MANETs. Multicasting is the transmission of data packets to more than one node sharing one multicasting address. The receivers form the multicast group. Actually, there could be more than one sender sending to a multicast group. Typical multicast protocols include On-Demand Multicast Routing Protocol (ODMRP) [5], Multicast Ad-Hoc on-Demand Distance Vector (MAODV) [6] and Ad hoc Multicast Routing (AMRoute) [7] and so on. Broadcasting is a special case of multicasting, which supports sending messages to all nodes in the network.

It is necessary to have reliable communications during large-scale emergency situations in today’s world. Information such as text, audio, or video may be broadcast or multicast to survivors to inform them of shelter locations, details of the disaster, how to respond to the disaster, and so on. Moreover, situational awareness data may be broadcast or multicast to various rescue teams such as the crisis center, police department, emergency medical services (i.e., hospitals, ambulances), and fire department. Considering the widely use of broadcast protocols and limited energy in MANETs, we focus on the study of some efficient broadcast protocols in MANETs and the design and evaluation of an energy-efficient broadcast protocol.

The goal of this project is to design and implement a new energy-efficient broadcast protocol and then compare it with three other broadcast protocols in MANETs. These three protocols are Classical Flooding, Simplified Multicast Forwarding (SMF) [8], and a coding-based broadcast protocol (CodeBCast) [9]. Classical Flooding is the simplest broadcast method, in which each node retransmits packets to all its neighbors if they are not duplicated. It is reliable for MANETs with low density nodes and high mobility, however, it is very harmful and unproductive as it may cause severe network congestion and quickly exhaust the nodes' energy. Although Classical Flooding has some disadvantages, given its widely used in the routing discovery in some protocols, we will use it as a benchmark for comparison. As an efficient multicast protocol, SMF [8] provides basic IP multicast forwarding for MANETs, which provides duplicate packet detection mechanisms for forwarding IP multicast packets and efficient relay set mechanisms for reducing contention and congestion in wireless multi-hop scenario. In additional, our proposed protocol is also implemented based on SMF, and thus through the performance comparison with SMF, we can know if our proposed protocol performs better. It is known that network coding in wired networks enables connections with rates that are higher than those achieved by plain routing only. However, the properties of wireless networks modify the formulation of time-varying network coding in a way that differs from the classical approach used in wired network coding. Since intermediate nodes are allowed to encode and decode messages they receive in network coding approach, less number of transmissions and receptions are needed to gather all the required packets, which make it energy-efficient. Considering network coding approach offers advantages in multicast and broadcast networks, such as high throughput and efficient energy, we study the network coding approaches in MANETs in recent years and select one network coding based broadcast protocol, CodeBCast [9], to compare with our proposed protocol. CodeBCast [9] applies the network coding approach, which is designed to achieve the maximum network throughput using a deterministic broadcast method, and resulting in a significant reduction in the number of transmissions in the network.

In Section 3, we review some energy-efficient protocols proposed in recent years, i.e. papers published by Guo and Yang which are all shown to be energy-optimized. However, most of them use the power-adjustable energy model or directional antenna, while our network is power-fixed and we use the omni-directional antenna. In the previous work focusing on the power-adjustable energy model, the power control ability at each node is exploited according to some information, i.e. by monitoring the distance in the received hello messages from all its children, and the transmission power is adjusted to the minimum level at each transmission in order to achieve the minimum energy consumption. Additionally, in the previous work focusing on the directional antenna propagation model, narrow beams is used in order to save energy because only the nodes located within the transmitter's antenna beam can receive the message, and the transmission power is usually adjusted based on the beamwidth in this type of network. Since the energy model is different, it does not make sense to compare our proposed protocol with these energy-optimized protocols.

Although SMF and CodeBCast are not energy-efficient protocols, they are proved to be efficient in the multicasting/broadcasting domain in terms of the packet delivery ratio and the network throughput. Moreover, the multipoint relay method in SMF and the network coding nature of CodeBCast make them consume less energy than the traditional routing. Through comparison of our protocol with them, we can see if they also perform well in terms of the energy consumption and network lifetime.

In recent research, there are mainly two groups of approaches used to develop energy-efficient broadcast routing protocols. In the first approach, each node can adjust its transmission power based on the distance to the receiving node(s) and the background noise either continuously or in a discrete fashion. In the other approach, all nodes in the network use a fixed power level for transmissions. In the energy model of our network, transmission power of each node is fixed. After a survey of the recent energy-efficient broadcast/multicast routing protocols, we find that few research on energy-efficient broadcast uses fixed transmission power of each node. Most of the energy-efficient protocols we mention in Section 3 are shown to be energy-optimized, but based on the power-adjustable model. Therefore, we focus on the energy-efficient broadcast routing protocol using fixed transmission power and propose a new energy-efficient broadcast protocol, EBOLSR, which extends the Energy-efficient OLSR (EOLSR) [10] protocol to a broadcast protocol.

In order to test and compare the performance of the above protocols, we use the network simulator, NS2 [11], to simulate two distinct scenarios, a WiFi scenario and a RiceanRescue scenario. Since Classical Flooding, SMF and CodeBCast have already been implemented by others, we only implement our own protocol, EBOLSR, and the detailed implementation is shown in Section 5. The performance of the four protocols are compared in terms of the packet delivery ratio, average end-to-end delay, energy consumption per packet, network lifetime and the average number of (E)MPRs per node of EBOLSR and SMF. The definitions of these metrics are given in Section 6.

The rest of the report is organized as follows. Section 2 reviews multicast/broadcast protocols in MANETs. Section 3 reviews the energy-efficient multicast/broadcast protocol in MANETs. Section 4 reviews the network coding approach in MANETs. Section 5 describes the design and implementation of our proposed protocol and other algorithms for comparison. Section 6 presents the simulation results. Section 7 concludes the report.

2 MULTICAST/BROADCAST PROTOCOLS

The multicast/broadcast services are critical in applications characterized by the close collaboration of teams with requirements for audio and video conferencing and sharing of text and images. Additionally, most routing protocols in MANETs rely on the broadcast function to exchange essential routing packets between mobile nodes and need the multicast function to make more efficient use of network bandwidth for some particular multimedia applications. Hence, broadcast and multicast are important operations for mobile nodes to construct a routing path in MANETs.

2.1 Multicast Protocols

Multicasting is the transmission of data packets to more than one node sharing one multicasting address. It is intended for group-oriented computing. Several multicast routing protocols have been proposed for MANETs, which can be classified as unicast-based, tree-based, mesh-based, or hybrid protocols, according to how distribution paths among group members are constructed.

2.1.1 Protocols Classification

- Unicast-based multicast protocols

  Some primitive broadcast/multicast protocols are just unicast-based. That is, for a source to send to N destinations, the protocol simply set up N unicast connections to achieve the function of multicast. Since few recent research focuses on this type of multicast protocols, we will not describe more about it, and will focus on the following two kinds of multicast protocols.

- Tree-based multicast protocols

  Tree-based multicast routing protocols can be further divided into source-tree-based and shared-tree based schemes, according to the number of trees per multicast group. In a source-tree-based multicast protocol, a multicast tree is established and maintained for each source node of a multicast group, and shared-tree-based multicast protocols use a single shared tree for all multicast source nodes. In the source-tree-based multicast protocol, each multicast packet is forwarded along the most efficient path, i.e. the shortest path, from the source node to each multicast group member, but this method incurs a lot of control overhead to maintain many trees. For the shared-tree-based multicast protocol, it has lower control overhead since it maintain only a single tree for a multicast group and thus is more scalable. Adaptive Demand-driven Multicast Routing (ADMR) [12] is source-tree-based and Multicast Ad Hoc On-Demand Distance Vector (MAODV) [6] is a shared-tree-based multicast protocol developed for MANETs.

- Mesh-based multicast protocols

  In mesh-based multicast protocols, more than one path exists between each sender and receiver. When a route fails, which is common in MANETs, there should be another route to deliver the data. Mesh-based multicast protocols support the redundancy of routes that provides fault tolerance. Obviously, this kind of protocol is more robust but less efficient since the mesh infrastructure also to be maintained and receivers typically receive more than one copy of a packet. On-Demand Multicast Routing Protocol (ODMRP) [5] is a mesh-based multicast protocol developed for MANETs.

- Hybrid multicast protocols

  A hybrid multicast protocol combines both the tree-based and mesh-based methods in order to achieve efficiency and robustness. It has two main procedures, the mesh creation and the tree creation. It first creates the virtual mesh links among the group members and a logic core will be selected from the members in this procedure. Then the mesh is used to establish the multicast tree which is initiated by the logical core. Ad hoc Multicast Routing (AMRoute) [7] is a hybrid multicast protocol developed for MANETs.

Multicast routing protocols can also be classified as proactive or reactive, depending on whether they keep routes continuously updated, or react on demand.

- Proactive protocols

  Proactive protocols attempt to find and maintain consistent, up-to-date routes between all source nodes and destination nodes regardless of whether these routes are needed. Periodic control messages are used to maintain routes up-to-date for each node. Examples include Ad hoc Multicast Routing protocol utilizing Increasing IdnumberS (AMRIS) [13] and Core-Assisted Mesh Protocol (CAMP) [14].

- Reactive protocols

  Unlike proactive protocols, reactive protocols create routes only when a source node requests them. Examples include the On-Demand Multicast Routing Protocol (ODMRP) [5] and the Multicast Ad Hoc on-Demand Distance Vector (MAODV) [6] protocol.

Since SMF is an efficient multicast protocol, we will simply mention it here. The specific operation of the protocol is in Section 5 and the detailed description and the feature of SMF is in the Appendix B.

2.1.2 Simplified Multicast Forwarding (SMF)

In MANETs, unicast routing protocols can provide effective and efficient mechanisms to flood routing control messages in the wireless routing area. For example, OLSR [4] provide distributed methods of dynamically electing reduced relay sets which can optimize flooding of routing control messages in the routing layer. Similarly, simpler multicast routing protocols that can optimize the forwarding of multicast traffic to all nodes in a routing area are also useful. One such solution is the Simplified Multicast Forwarding (SMF) specification designed within the Internet Engineering Task Force (IETF) [8]. Considering the multicast efficiency of SMF, we decide to select it to compare the performance with our proposed protocol in the Section 6.

SMF extends the efficient flooding concept to the data forwarding plane for IP multicast packets, which provides an appropriate multicast forwarding capability. More detailed protocol description is in the Appendix B.

2.2 Broadcast Methods

Broadcasting is the process in which a source node sends a message to all other nodes in the network, and it is also a special case of multicasting. Since even unicast and multicast routing protocols often have a broadcast component, broadcasting is important in MANETs. For instance, protocols such as DSR [1], AODV [2], Zone Routing Protocol (ZRP) [15] and Location Aided Routing (LAR) [16] use broadcasting to establish routes. Broadcasting methods have been categorized into four families utilizing the IEEE 802.11 MAC specifications [17]:

a) Classical Flooding
b) Probability-Based Methods
c) Area-Based Methods
d) Neighbor Knowledge Methods

2.2.1 Classical Flooding

In Classical Flooding, a source node broadcasts a message to all its neighbors, and each of these neighbors will check if they have seen this message before. If yes, the message will be dropped; otherwise the message will be rebroadcast at once to all their neighbors. The process goes on until all nodes received the same message. Although this method is very reliable for MANETs with low density nodes and high mobility, it is very harmful and unproductive as it may cause severe network congestion and quickly exhaust the nodes' energy. Classical Flooding is the simplest case of SMF multicast forwarding we will mention later.

2.2.2 Probability-Based Methods

The probability-based method tries to solve the problem of the Classical Flooding method. Each node $i \in N$ is given a predetermined probability p_i for rebroadcasting. Thus, the network congestion and collisions can be minimized if some nodes do not rebroadcast. In this approach, there is a danger that some nodes will not receive the broadcast message. Probabilistic Scheme and Counter-Based Scheme are both probability-based methods which were proposed by [18].

a) Probabilistic Scheme

   The Probabilistic Scheme is similar to flooding, except that nodes only rebroadcast with a predetermined probability. In dense networks, multiple nodes share a similar transmission coverage. Thus, randomly having some nodes that will not rebroadcast can save network resources. In sparse networks, there is much less shared coverage, thus, nodes cannot receive all the broadcast packets with the Probabilistic Scheme unless the probability parameter is high. When the probability is 100%, this scheme is identical to Classical Flooding.

b) Counter-Based Scheme

In Counter-Based Scheme, when a node tries to rebroadcast a packet, the packet may be blocked by the busy medium, backoff procedure and other queued packets. During this period, a node may receive the same packet from other nodes before the queued packet is sent out. A counter c is used to record the number of times the broadcast packet is received. A counter threshold C is chosen. When $c>=C$, stop the rebroadcasting. Otherwise, the packet should be rebroadcasted.

2.2.3 Area-Based Methods

Area-based methods assume nodes have common transmission distances. A node will rebroadcast only if the rebroadcast will reach a sufficient additional coverage area. Distance-Based Scheme and the Location-Based Scheme proposed also by [18] are both area-based methods.

a) Distance-Based Scheme

In the Counter-Based Scheme, a counter is used to decide whether to rebroadcast or not. In the Distance-Based Scheme, the distance between nodes is used to make the decision. Since the distance is related to the coverage area (($S = \pi d^2$), where S is the coverage area and d is the distance), it can be used as a metric to determine whether to rebroadcast or not. Suppose that before sending a broadcast packet, node N has received the same packet several times. Let d_{min} be the distance to the nearest node from which the packet is received and D be some distance threshold. If $d_{min} <D$, N does not rebroadcast the packet; otherwise, the packet should be rebroadcasted.

b) Location-Based Scheme

In the Location-Based Scheme, each node needs to establish its own location in order to estimate the additional coverage more precisely. This approach can be supported by the global positioning system (GPS). Each node will add its own location to the header of each message it broadcasts. When a node receives a message, it first checks the location of the sender and computes the additional coverage area to rebroadcast. If the additional coverage area to rebroadcast is less than a given threshold, the message is dropped, otherwise the message will be rebroadcast.

2.2.4 Neighbor Knowledge Methods

In a Neighbor Knowledge Method, neighborhood information is maintained, and is used to determine whether to rebroadcast or not. Methods include Flooding with Self Pruning [19], Dominant Pruning [19], Multipoint Relaying [20], Scalable Broadcast Algorithm (SBA) [21], Ad Hoc Broadcast Protocol (AHBP) [22], Connected Dominating Set (CDS)-Based Broadcast Algorithm [23] and the Lightweight and Efficient Network-Wide Broadcast (LENWB) protocol [24].

a) Self Pruning [19]

   In Self Pruning, each node is required to have knowledge of its neighbors by periodically exchanging Hello messages, which include the list of the sending node’s neighbors in the packet header. Upon receiving the Hello message, the node will compare its neighbor list with the sender’s list. It will rebroadcast if additional nodes could be reached, otherwise it will drop the message. This is the simplest approach in the neighbor knowledge method, but there is still some message redundancy in this method.

b) Dominant Pruning [19]

   In Dominant Pruning, each node learns its neighbor knowledge within 2 hops via Hello messages. When rebroadcasting the packet, each node will choose some or all of its 1-hop neighbors as rebroadcasting nodes. When a node receives a broadcast packet, it checks the header to see if its address is part of the list. If so, it uses a Greedy Set Cover algorithm [19] to determine which neighbors should rebroadcast the packet, given knowledge of which neighbors have already been covered by the sender’s broadcast. The Greedy Set Cover algorithm recursively chooses 1-hop neighbors which can cover the most 2-hop neighbors and recalculates the cover set until all 2-hop neighbors are covered.

c) Multipoint Relaying [20]

Like Dominant Pruning, rebroadcasting nodes in Multipoint Relaying are also chosen by upstream senders. Each node pre-selects some or all of its 1-hop neighbors to rebroadcast the packets it sends to them. The chosen nodes are called Multipoint Relays (MPRs) and the node that selects MPRs is called the MPR selector. MPRs are the only nodes allowed to rebroadcast a packet received from the MPR selector. They are alternately called “forwarder” mentioned later in Section 5. Each MPR also chooses its own MPRs from its 1-hop neighbors as well. Since each node knows the network topology within 2 hops, it can select 1-hop neighbors as MPRs that most efficiently reach all nodes within the 2-hop neighbors. [20] proposed the algorithm for a node to choose its MPRs.

Multipoint Relaying is described in detail as part of the Optimized Link State Routing (OLSR) protocol defined by an Internet RFC [4]. In OLSR, each node can find its 1-hop and 2-hop neighbors through exchanging Hello messages, and select MPRs from all 1-hop neighbors which cover the most 2-hop neighbors. OLSR uses Topology Control (TC) and Host Network Address (HNA) messages along with MPR forwarding to disseminate routing information throughout the network.

3 ENERGY-EFFICIENT MULTICAST/BROADCAST PROTOCOLS

Since each node in MANETs typically draws energy from a battery with finite capacity, energy efficiency is an important design consideration for MANETs. In this section, we will describe the energy efficiency problem in broadcast and multicast routing protocols and then propose the energy-efficient broadcast protocol we designed. In recent research, there are mainly two fundamental classes of energy-aware broadcast/multicast problems [27]: the minimum energy broadcast/multicast (MEB/MEM) problem and the maximum lifetime broadcast/multicast (MLB/MLM) problem. The MEB/MEM problem aims to minimize the total energy consumption and the MLB/MLM problem aims to maximize the network lifetime which is usually defined as the time to the first node failure due to battery exhaustion. In recent research, solutions to MEB/MEM and MLB/MLM problems can be classified into several categories, i.e. dynamically adjusting the transmission power of wireless nodes, reducing the volume of information transferred by means of aggregation, making nodes sleep in order to spare energy, using energy-efficient routing, and so on. Since the beam-forming technology can provide energy saving in recent research, some energy-efficient protocols use directional antennas which reduce the radio interference and thus enable greater information capacity in the network. Omni-directional antenna is only a special case of directional antenna (when the coverage angle θ=360 degree) and therefore many directional algorithms also apply to omni-directional.

3.1 MEB/MEM

The energy optimization problem [28, 29] was first proposed in broadcasting for omnidirectional antenna MANETs, and then some results [30, 31, 32, 33, 34, 35] have been extended to multicasting. A lot of work has been proposed for the problems of minimizing the energy consumption for broadcasting and multicasting in wireless ad hoc networks, which can be classified into the MEB problem and the MEM problem. We first review the MEB/MEM problem with omni-directional antennas, and then describe the latest development to directional antennas.

3.1.1 MEB/MEM with Omni-directional Antennas

The MEB problem in MANETs with omnidirectional antennas has been shown to be NP-complete [28]. To solve this problem, one approach is to obtain globally optimal solutions, i.e. mixed integer linear programming (MILP), and the other approach is heuristics. The heuristic algorithms proposed recently for the MEB problem can be classified as follows [27]:

- Spanning tree algorithms: A spanning tree of a connected graph *G* can be defined as a maximal set of edges of *G* that contains no cycle, or as a minimal set of edges that connects all vertices. Most energy-efficient protocols using spanning tree algorithms build source-based broadcast trees which start from the source node and is expanded until all nodes in the network are covered according to some objective function. Examples include minimum spanning tree (MST) [30], shortest path tree (SPT) [30], broadcast incremental power (BIP) [30] and so on.

- Topology control algorithms: Topology control algorithms are usually used in the power adjustable network. Each node can adjust its transmission power and select certain neighbors for communication, and thus certain objective functions of the transmission powers are optimized. Examples include relative neighborhood graph (RNG) [36], local minimum spanning tree (LMST) [37], incident MST [38] and so on.

- Local search algorithms: This algorithm uses some power-related selection method to form a new broadcast topology iteratively from an initial broadcast topology so that the necessary connectivity properties are maintained. The final topology must have a lower power than the initial feasible topology. Examples include the tree based algorithms Sweep [27], IMBM (iterative maximum-branch minimization) [39], and broadcast incremental-decremental power (BIDP) [40] and so on.

For the MEM problem, the heuristic algorithms can be classified into the following three groups [27]:

- Pruning: MEM problem was studied using the same approach as the MEB problem.

The only difference is that the final minimum energy multicast tree is obtained by pruning the minimum energy broadcast tree. Examples include pruned minimum spanning tree (P-MST) [30], pruned shortest-path tree (P-SPT) [30], and pruned broadcasting incremental power (P-BIP) [30] and so on.

- Minimum Steiner tree: A minimum Steiner tree is a spanning tree with weight less than or equal to the weight of every other spanning tree. Since finding a minimum Steiner tree is NP-complete, some fast heuristics can find near-minimum trees with constant approximation ratio and can be applied for the MEM problem directly [41]. Examples include shortest path first (SPF) [42], the minimum incremental path first (MIPF) [42] and so on.

- Local search: Similar to the approach for the MEB problem, a local search method is used to improve an initial feasible multicast topology iteratively. Examples include refining energy-efficient source-shared multicast tree (S-REMiT) [43], multicast incremental- decremental power (MIDP) [40] and so on.

3.1.2 Algorithm Requirement and Latest Development

Beam-forming is a general signal processing technique used to control the directionality of the transmission and reception of radio signals. It directs signals more efficiently toward a specific station in wireless networks by monitoring interference. Since the beam is generated only toward a certain direction, it creates less interference, less energy consumption and more information capacity in the network. As a result, the use of directional antennas has great potential in energy-constrained wireless ad hoc network.

The optimization of the algorithms (i.e., either energy minimization or lifetime maximization) also involves algorithms that are computational complex, and therefore cannot meet the real-time requirement of most Internet traffic. Many existing heuristics cannot handle a very large network, not to mention their optimality cannot be proven (one reason of calling it a heuristics). So optimization with low-complexity (meeting real-time requirement) is a continuing research topic.

Wieselthier et al. [44] first studied the MEB/MEM problems considering both the energy conservation with directional antennas and the wireless multicast advantage property for broadcasting/multicasting. Two heuristic algorithms called the Reduced Beam BIP/Reduced Beam MIP (RB-BIP/RB-MIP) algorithm and the Directional BIP/Directional MIP (D-BIP/D-MIP) algorithm were proposed as the extensions of the BIP/MIP algorithm for the situation of using adaptive antennas. The RB-BIP/RB-MIP Unlike BIP/MIP, RB-BIP/RB-MIP reduces the antenna beamwidth at each node to fit the minimum possible angle to cover all its child nodes after the BIP/MIP tree is constructed, and D-BIP/D-MIP utilizes wireless multicast advantage property in the core of the algorithm while building a routing tree. However, no optimization is realized.

Inspired by RB-BIP/RB-MIP and D-BIP/D-MIP, [45] developed a general analytical Mixed Integer Linear Programming (MILP) model for the MEM problem with adaptive antennas. Simulation results showed that an optimal solution of the MEM problem using this model can always be obtained in a timely manner for moderately sized networks. [45] also proposed two polynomial time algorithms, Directional Multicast Incremental-Decremental Power (D-MIDP) and Reduced Beam Multicast Incremental-Decremental Power (RB-MIDP) to handle large networks. D-MIDP iteratively reconstructs the multicast tree in the direction where can maximize the total energy saving using global topology information, and the beamwidth constraint is also taken into account at each improvement step. The only difference between RB-MIDP and D-MIDP is that, the initial broadcast tree is constructed using D-BIP in D-MIDP and is constructed using RB-BIP in RB-MIDP. Simulation results showed that D-MIDP and RB-MIDP have a time complexity of $O(n^3)$, which is a bit lower than that of D-MIP and RB-MIP. RB-MIDP/D-MIDP also provides much better performance than RB-MIP/D-MIP in directional antenna applications and also outperforms MIP in omnidirectional antenna applications.

A constraint formulation for the joint optimization problem of MEM and Antenna Orientation in terms of MILP is discussed by Guo and Yang [46]. The optimal solution can be solved by MILP in a timely manner for moderately sized networks with switched antennas. Considering the excessive time consumption to solve the joint optimization problem based on the MILP model, they also proposed two polynomial time heuristic algorithms, Beam-Shifting MST (BS-MST) and Beam-Shifting Multicast Incremental Power (BS-MIP), and a general post-process operation, Tree Reconstruction (TR), for handling larger networks. The BS-MST algorithm is based on the MST algorithm in which nodes selected to be added into the tree must satisfy the antenna orientation requirement. In BS-MIP algorithm, the incremental cost of adding a new node into a tree involves simply increasing transmission range, shifting the antenna beam, or a combination of the two operations. TR iteratively reconstructs the multicast tree based on the maximal decremental power by switching a tree arc with a nontree arc until no more power saving can be obtained. Simulation results showed that the TR operation significantly improves the performance of both heuristic algorithms.

3.2 MLB/MLM

The network lifetime can be defined in several ways, such as the time to the first node failure due to battery outage, the time to the unavailability of an application functionality, or the time to the first network partitioning. However, most researchers consider the network lifetime as the time to the first node failure [47].

The MLB/MLM problem has been explored using similar heuristic approaches as the minimum energy problem. Some previous work, i.e. [48], was proposed in broadcasting for omnidirectional antenna MANETs, and then some results [49, 50, 51, 52, 53] have been extended to multicasting. A lot of work has been proposed for the problems of maximizing the network lifetime for broadcasting and multicasting in wireless ad hoc networks, which can be classified into the MLB problem and the MLM problem. We first review the MLB/MLM problem with omni-directional antennas, and then describe the latest development to directional antennas.

3.1.3 MLB/MLM with Omni-directional Antennas

A special case of the MLB problem has been studied first in [54], where the ad hoc network is model as an undirected graph. The conclusion in [54] is that if the link weight is defined as the reciprocal of the link lifetime, i.e. $w_{vu} = \frac{1}{t_{vu}} = \frac{p_{vu}}{\varepsilon_v}$, where p_{vu}

represents the power required for data transmission from node *v* to node *u*, t_{vu} represents the maximum time that the arc *(v, u)* could last before battery depletion in node *v*, and ε_v is the residual battery energy at node *v*. The graph is undirected, and the minimum spanning tree of the graph has the maximum lifetime for the broadcast. This special case assumes a symmetric propagation model and identical battery energy supply at each node in the network. Several polynomial-time algorithms, minimum decremental lifetime (MDLT) [47], directed minimum spanning tree (DMST) [55], and directed prim broadcast tree (DPBT) algorithm [55] have been proposed. All of them are essentially implementations of Prim's algorithm running on a general directed graph.

Similar to the MLB problem, the special case of the undirected graph can be solved by pruning a minimum spanning tree as proved in [56]. For the general directed graph, [49] proved that the MLM problem is also NP-complete. The most advanced results for the general case are the directed prim multicast tree (DPMT) algorithm [56], the bottleneck multicast tree (BMT) algorithm [57], and the min-max tree (MMT) algorithm [58].

3.1.4 Algorithm Requirement and Latest Development

As discussed in Section 3.1.2, there is the desire of optimization with low-complexity which remains to be a continuing research topic.

To this goal, the static-weight DPMT (S-DPMT) and the dynamic-weight DPMT (D-DPMT) algorithm [56] are two algorithms respectively for both omni-directional antennas and adaptive antennas. Both S-DPMT and D-DPMT are extension of the DPMT [48] algorithm. S-DPMT directly applies DPMT, assuming that the transmitting antennas are omni-directional. After the tree is constructed in this manner, each internal node reduces its antennas beamwidth to the smallest possible value that provides the coverage of the node's downstream neighbors in the tree. Unlike the pre-computed and unchanged input link weights in DPMT, D-DPMT must dynamically update the weights for each link after each step of including a new node into the multicast tree to reflect the value of beamwidth. Simulation results showed that D-DPMT performs much better than the other algorithms like RB-MIP-β and D-MIP-β (β = 0, 1, 2). On the average, D-DPMT has about double the lifetime for all possible minimum beamwidth when compared to D-MIP.

The MLB/MLM problem for directional antenna scenarios has been proven a NP-complete problem [59]. The only exact solution is the MILP formulation presented in [60], where a new algorithm called Maximum Lifetime Routing for Multicast with Directional antennas (MLR-MD) was proposed. The basic idea of the MLR-MD algorithm is to start with a multicast routing solution, and then iteratively improve the network lifetime by identifying nodes with the shortest lifetime and revising routing topology and the corresponding beam-forming behavior. Simulation results showed that MLR-MD provides consistent performance improvement over the D-MIP algorithm and the time complexity of the algorithm is strictly polynomial.

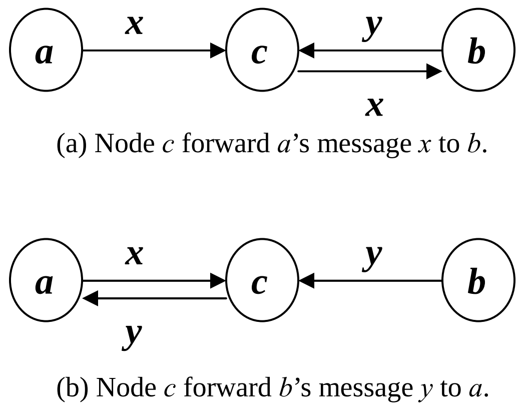
[61] proposed the MMT for Directional Antenna (MMT-DA) algorithm, which is a further extension using dynamic link-weight update in the tree construction. Similar to the distributed MMT algorithm [58], nodes are added into the tree by grow-and-search cycles in MMT-DA. MMT-DA provides much better performance than other algorithms in terms of the multicast lifetime. Its superiority is even greater in network examples with larger minimum beamwidth, and very close to the optimum obtained by exact method [58]. If the weight of the bottleneck link is well estimated, the algorithm has linear expected message complexity, while for the general case, the communication complexity is of the order $O(n^2)$.

4 NETWORK CODING

In the general model of data networks, recent research in information theory discovered that routing alone is not sufficient to achieve maximum information transmission rates [62]. Additional encoding and decoding operations at relay nodes between source and destinations are in general necessary for an optimal transmission strategy. Such coding operations are referred to as network coding. Due to the mechanism of network coding approach, less number of transmissions and receptions are needed to gather all the required packets, which make it energy-efficient. Thus, we will pick a network coding-based protocol to compare with our proposed energy-efficient protocol.

4.1 Description of Network Coding

The following example demonstrates the basic idea and potential benefit of network coding in wireless networks. Assume node *c* gets two messages from nodes *a* and *b* respectively. In order to let *a* and *b* receive each other's message, *c* needs to forward both messages as a traditional forwarding node, shown in Figure 1. With network coding, *c* only needs to forward one coded message containing both original messages, combined through the XOR operation, and *a* and *b* can decode the message with the help of their own messages through the XOR operation, shown in Figure 2. Note that network coding works only when there are multiple messages broadcast at the same time in the network.



(a) Node *c* forward *a*'s message *x* to *b*.

(b) Node *c* forward *b*'s message *y* to *a*.

Fig. 1 Messages forwarding without network coding.

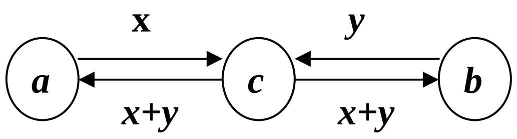


Fig. 2 Messages forwarding with network coding.

The example shown above uses a simple XOR operation for the packet combination. Next, a description of the more general random linear network coding approach is given.

Suppose that, at the source node *s*, we have *K* message packets $c_1, c_2, \ldots, c_K$. The message packets are initially present in the memory of node *s*. The coding operation performed by each node is simple to describe and is the same for every node: Received packets are stored into the node's memory and packets are formed for injection with random linear combinations of its memory contents. The coefficients of the combination are drawn randomly from the finite field *Fq*. Since all coding is linear, we can write any packet *x* in the network as a linear combination of $c_1, c_2, \ldots, c_K$, namely, $x = \sum_{k=1}^{K} \alpha_k c_k$. We call α = $(\alpha_1, \alpha_2, \ldots, \alpha_k)$, the global encoding vector of *x*, and we assume that it is sent along with *x*, in its header. A sink node collects packets and, if it has *K* packets with linearly independent global encoding vectors, it is able to decode and recover the original message packets. Decoding can be done by Gaussian elimination.

Ahlswede et al. [63] showed the utility of network coding for multicast. Koetter and Medard [64] then showed that codes with a simple, linear structure were sufficient to achieve the capacity of multicast connections in lossless networks. This result was augmented by [65], which showed that, in fact, a random construction of the linear codes was sufficient. The utility of such random linear codes for reliable communication over lossy packet networks, such as wireless ad hoc networks, was soon realized [65]. In [66], a prescription for the efficient operation in MANETs was given, which proposed using the random linear coding scheme of [66] coupled with optimization methods for selecting the time and locations for injecting coded packets into the network.

4.2 Network Coding in MANETs

It is known that network coding in wired networks enables connections with rates that are higher than those achieved by plain routing only. However, the properties of wireless networks (e.g., omnidirectional transmissions, destructive interference, single transceiver per node, finite energy) modify the formulation of time-varying network coding in a way that differs from the classical approach used in wired network coding. To enable a practical application of network coding to MANET, we need to take into account several problems, such as subgraph selection, wireless medium contention, coding efficiency, complexity of encoding and decoding and the energy efficiency. The description of each problem is given below.

4.2.1 Subgraph Selection

The problem of selecting the time and locations for injecting coded packets is called subgraph selection.

In [67], a prescription for the efficient operation in MANETs was given, which is proposed using the random linear coding scheme coupled with optimization methods for selecting the time and locations for injecting coded packets into the network. The network can be modeled with a directed hypergraph $H = (N, A)$, where N is the set of nodes and A is the set of hyperarcs. A hyperarc is denoted by (i, J), where the start node i, is an element of N and the set of end nodes J, is a non-empty subset of N.

A_{iJK}— the counting process which describes the arrival of packets that are injected on hyperarc (i, J) and received by the set of nodes $K \subset J$, i.e. for $\tau \geqslant 0$

$A_{iJK}(\tau)$ — the total number of packets received between time 0 and time τ by all nodes in K due to (i, J)

z_{iJK}— the average rate of A_{iJK}

z— the coding subgraph, a rate vector

$$z_{iJK} = \lim_{\tau \to \infty} \frac{A_{iJK}(\tau)}{\tau} \qquad (1)$$

Given a coding subgraph z, it is shown in [65] that a multicast rate arbitrarily close to R packets per unit time from source node s to destination nodes in the set T, is achievable with coding from the source node to destination nodes if and only if there exists a flow vector $x^{(t)}$ satisfying:

$$\sum_{\{J|(i,J\in A)\}} \sum_{j\in J} x^{(t)}{}_{iJj} - \sum_{\{j|(j,I\in A,i\in I)\}} x_{jIi}{}^{(t)} = \sigma_i{}^{(t)} \qquad (2) \qquad \text{for all } \; i \in N$$

$$\sum_{j\in K} x_{iJj}{}^{(t)} \le \sum_{\{L\subset J|L\cap K\neq\phi\}} z_{iJL} \qquad (3) \qquad \text{for all } \; (i,J)\in A \; \text{ and } \; K\subset J\,,$$

where

$$\sigma_i{}^{(t)} = \begin{cases} R & i=s \\ -R & i=t \\ 0 & o.w. \end{cases}$$

The subgraph selection problem is the problem of finding a coding subgraph z of minimum cost satisfying (2) and (3) [67]

$$\text{minimize } f(z) \quad \text{subject to } \; z\in Z$$

$$\sum_{j\in K} x_{iJj}{}^{(t)} \le \sum_{\{L\subset J|L\cap K\neq\phi\}} z_{iJL} \qquad \forall (i,J)\in A, K\subset J, t\in T$$

$$\sum_{\{J|(i,J\in A)\}} \sum_{j\in J} x^{(t)}{}_{iJj} - \sum_{\{j|(j,I\in A,i\in I)\}} x_{jIi}{}^{(t)} = \sigma_i{}^{(t)} \qquad \forall i\in N, t\in T$$

$$x_{iJj}{}^{(t)} \ge 0 \qquad \forall (i,J)\in A, j\subset J, t\in T$$

The prescription given in [67] potentially allows for the optimal way of setting up a single connection to be found. But finding an optimal solution may be complex, especially under the complex constraints imposed by MANETs.

4.2.2 Wireless Medium Contention

In wireless environments, network coding relies on the broadcast nature of the medium to deliver a single encoded packet to multiple receivers. However, IEEE 802.11 broadcast has no collision detection or avoidance mechanism. As a result, broadcast works badly in congested environment where the collision probability is high.

[68] proposed a general modeling and solution framework for the throughput optimization problem in wireless networks. In their framework, data routing, wireless medium contention and network coding were jointly considered to achieve the optimal network performance. [69] introduced a novel scheduling approach named Popularity Aware Scheduling (PAS) for network-coding-based MANET. Popularity refers to how

much one new data block is wanted by neighboring nodes and can be used to make the content distribution more efficient by assigning different levels of channel access priorities to blocks at the MAC layer. The higher the popularity value, the higher a priority was assigned. Performance evaluations indicated that the cooperation of network coding and MAC layer scheduling was both necessary and efficient.

4.2.3 Coding Efficiency

As we know, broadcasting is the simplest operation used in MANETs for the dissemination of data and control messages in many applications. In order to avoid the broadcast storm problem caused by simple blind flooding, only selected nodes forward data to the entire network. CDS is a typical solution, but finding a minimum CDS is a NP-complete problem [70]. Since network coding allows intermediate nodes to combine packets before forwarding, the total number of transmissions with the help of network coding can be reduced, compared to broadcasting using the same forwarding nodes without coding.

Both probabilistic [18] and deterministic [21] approaches have been proposed for efficient broadcast. Probabilistic approaches use limited neighborhood information and require relatively high broadcast redundancy to maintain the packet delivery ratio. Deterministic approaches select a few forwarding nodes to achieve high delivery and most are localized, where each node determines whether to forward or not based on its h-hop neighborhood information. In [71], a proactive compensation packet is periodically broadcast, which is constructed from unforwarded messages using network coding to improve the delivery ratio of the probabilistic broadcast approach. [9] showed how network coding can be applied to a deterministic broadcast approach, resulting in significant reductions in the number of transmissions in the network. The authors proposed two algorithms, which rely only on local 2-hop topology information and made extensive use of opportunistic listening to reduce the number of transmissions. In [72], each node equipped with directional antennas can divide the omnidirectional transmission range into several sectors and turns some of them on for transmission. In the proposed scheme using a directional antenna, forwarding nodes selected locally only need to transmit broadcast messages, original or coded, to restricted sectors. Simulation results showed that the number of transmissions can be greatly reduced.

4.2.4 Complexity of Encoding and Decoding

Traditional network coding uses operations over large finite fields. Decoding operations have quadratic complexity and further encoding operations are also complicated since they involve multiplications in large finite field. These make their use in high throughput applications questionable.

[73] classified the links in a network into two categories: 1) links entering relay nodes, and 2) links entering destination nodes. They showed that the same multicast capacity can be achieved by applying network coding only on the links entering relay nodes, which leads to a saving in the processing and implementation complexity.

4.2.5 Energy Efficiency

Since nodes have limited power in MANETs, energy efficiency is a critical design parameter.

[74] characterized the minimum amount of energy required to transmit one unit of information from a source node to all receivers, and developed distributed algorithms that allow to approach the optimal performance in practice. [75] showed that under a simplified layered model of wireless networks, the minimum-energy multicast problem in MANETs was solvable as a linear program, assuming network coding. Compared with conventional routing solutions, network coding not only promised a potentially lower energy per bit, but also enabled finding the optimal solution in polynomial time, in sharp contrast with the NP-complete problem of constructing the minimum-energy multicast tree as the optimal routing solution [76].

5 ENERGY-EFFICIENT BROADCAST OLSR (EBOLSR) and Other Algorithms for Comparison

As mentioned in Section 1, the goal of our project is to design and implement a new energy-efficient broadcast protocol in MANETs, and then compare the performance with three other broadcast protocols, Classical Flooding, SMF and CodeBCast. In the following, we will present our proposed protocol in Subsection 5.1. The procedure of other protocols are presented in Subsection 5.2 (Classical Flooding), Subsection 5.3 (SMF), and Subsection 5.4 (CodeBCast).

Since EBOLSR extends the OLSR protocol which uses the multipoint relaying method, the broadcast algorithm in our protocol belongs to the neighbor knowledge method mentioned in Section 2.4. From the energy perspective, we adapt the multipoint relay selection algorithm in EOLSR which aims to maximize the network lifetime, and thus our protocols belongs to the MLB problem mentioned in Section 3.2.

5.1 EBOLSR

5.1.1 Motivation

As mentioned in Section 3, several kinds of solutions can be used to maximize the network lifetime. In this project, we use an energy-efficient routing for the following reasons:

- In the energy model of our network, transmission power of each node is fixed.
- In our network, all information needs to be delivered, such as push-to-talk voice streams, and nodes are on all the time.
- Broadcasting requires more nodes in the network to participate in the transmission, so reducing the number of rebroadcasting nodes can optimize network flooding, reduce the energy consumed and increase the network lifetime.
- Multi-hop transmissions are energy consuming and reducing the energy spent in the transmission of a packet from its source to all nodes increases network lifetime.
- Most of the work in the fixed power approach aims at reducing the overall number of retransmissions during a broadcast. However, most existing work in this field only considers how to save energy with the minimum relaying set, but does not take

nodes' residual energy into account. As far as we know, few research on energy-efficient broadcast focused on fixed transmission power of each node, we therefore propose and implement a new energy-efficient broadcast routing protocol.

The OLSR routing protocol has been standardized by the IETF [4]. However, this protocol does not take energy into account. [10] proposed a new energy-efficient unicast routing protocol, EOLSR, which extends OLSR in order to make it energy efficient. EOLSR is designed to maximize the network lifetime by selecting the path with the minimum cost, where the cost takes into account the residual energy of each visited node and the energy consumption of a packet on this path. Based on the multipoint relaying strategy in EOLSR, we propose a new energy-efficient broadcast protocol, EBOLSR, which aims at increasing the network lifetime for broadcast communication.

5.1.2 Energy Consumption Model

Each node's radio can be in one of the following three states:

- Transmitting : node is transmitting message with transmission power P_t,
- Receiving: node is receiving message with reception power P_r,
- Idle: when no message is being transmitted, the nodes stay idle and keep listening the medium with P_{idle},

Since transmission is more expensive than receiving, and nodes in idle state consume the least energy, we therefore have the following power condition:

$$P_{idle} < P_r < P_t$$

Each state operates at different power levels and that level is fixed for all nodes in the network. When a transmitter transmits one packet to next hop, because of the shared nature of wireless medium, all its neighbors receive this packet even if it is intended to only one of them.

5.1.3 Hello Message Format

Since the energy is taken into account for the relaying node selection in our protocol, the residual energy of each node in the network should be recorded and exchanged with others. A Hello message is emitted every Hello interval for link sensing, neighborhood detection and relaying node selection signaling. Rather than propagating residual energy information with a new message, given the network overhead and efficiency, we add some fields in the Hello message which is used for recording the residual energies of the sending node and all its 1-hop neighbors. The code of the Hello message modification is shown in the Appendix A. Each Hello message has several fields as follows:

- Reserved: should be set to all 0s
- HTime: specifies the Hello emission interval
- Residual Energy: specifies the residual energy of the node
- Link Code: specifies information about the link between the interface of the sender and neighbors' interfaces
- Link Message Size: specifies the size of the link message
- Neighbor Interface Address: specifies the address of an interface of a neighbor node
- Neighbor Residual Energy: specifies the residual energy of 1-hop neighbors

Through exchanging Hello messages periodically, each node learns which node has a link with it and decides which nodes are its 1-hop and 2-hop neighbors. For any node in the network, *N*, *N*'s 1-hop neighbors are nodes from which Hello messages are received. *N*'s 2-hop neighbors are the 1-hop neighbors of the sending nodes, but not also *N*'s current 1-hop neighbors. With such knowledge, each node can compute and select its relaying nodes, according to the following EMPR selection strategy.

5.1.4 Energy-Efficient Selection of MPRs

Different from MPRs in OLSR, the energy-efficient selection of MPRs is the multipoint relay selection according to the residual energy, and the selected nodes are denoted EMPRs. In EOLSR, three variants depending on the neighbors considered for the computation of the minimum residual energy are discussed, where $E_R(M)$ denotes the residual energy of node *M*: [10]

a) The E strategy considers only $E_R(M)$, the residual energy of the EMPR candidate, *M*;

b) The M1E strategy considers the weighted residual energy of the EMPR candidate *M* and its 1-hop neighbors: $\min(\frac{E_R(M)}{P_{trans}+P_{rcv}}, \min_{D\in 1hop(M)}(\frac{E_R(D)}{2\times p_{rcv}}))$. The weights of $E_R(M)$ and $E_R(D)$ take into account the role played by the nodes *M* and *D* in a reansmission from *N*, the node performing the EMPR selection, to *D*, one of its 2-hop neighbors, via the node *M*. It represents the maximum transmission duration that can be sustained;

c) The M1E strategy considers the weighted residual energy of the EMPR candidate *M* and its 1-hop and 2-hop neighbors: $\min(\frac{E_R(M)}{P_{trans}+P_{rcv}}, \min_{D\in 1hop(M)}(\frac{E_R(D)}{2\times p_{rcv}}), \min_{D\in 2hop(M)}(\frac{E_R(D)}{P_{rcv}}))$.

In OLSR, MPRs are selected from the node's 1-hop neighbors that can cover the most number of its 2-hop neighbors based on its node degree. However, in EOLSR, any of the three variants described above can be used as the selection criterion in the EMPR selection algorithm. [10] evaluated the average number of EMPRs per node as a function of network density with the three selection variants, and compared it with the average number of MPRs per node in OLSR. Simulation results showed that the number of EMPRs per node is higher than the number of MPRs per node. Furthermore, the M1E and M2E selection strategies give comparable results. [10] also compared the network lifetime obtained with the three selection strategies. Simulation results showed that M1E and M2E gave the same result and both outperformed the E selection strategy. Since M1E is less complicated to compute and needs less information from the network than M2E, we will use the M1E selection strategy in our proposed protocol to select EMPRs.

After exchanging Hello messages periodically, each node has its own one-hop neighbor and two-hop neighbor set. The EMPR selection algorithm by any node, *N*, is described as follows:

a) Calculate the maximum transmission duration that can be sustained, $\min(\frac{E_R(M)}{P_{trans}+P_{rcv}}, \min_{D\in 1hop(M)}(\frac{E_R(D)}{2\times p_{rcv}}))$, where $E_R(M)$ is the residual energy of *N*'s 1-hop neighbor, and $E_R(D)$ is the residual energy of *N*'s 2-hop neighbor.
b) Calculate *D(M)*, the degree of 1-hop neighbors of *M*, excluding the 1-hop neighbors of *N*.
c) Sort the 1-hop neighbors in decreasing order of the selection criterion in step a).

d) Select the first 1-hop neighbor, add it to the EMPR set, and then remove its 1-hop neighbors in *N*'s 2-hop neighbor set.
e) Select the next 1-hop neighbor in order and repeat step d), until all nodes in *N*'s 2-hop neighbor set are removed. If there is more than one node with the same maximum transmission duration, select the node with the highest *D(M)*.
f) The EMPR set is generated. If the 1-hop neighbors of any node in the EMPR set are a subset of other nodes' 1-hop neighbors currently in the EMPR set, remove this node from the EMPR set.

For the code of EMPR selection, see the Appendix A.

5.2 Classical Flooding

As the simplest broadcasting method, Classical Flooding only provides the following two mechanisms:

- Every node retransmits every packet heard once and only once,
- Duplicate packet detection is performed and is the critical feature used to avoid additional retransmissions.

To implement this protocol, Each packet is uniquely identified by the address of the sender and a sender-assigned ID. In many cases, the latter would be sufficient for identification purposes within NS2, but in real implementations, such IDs are probably only unique within a sender, so the approach of adding the sender ID/IP address was chosen. The algorithm is described as follows:

a) Upon receiving a packet, the receiver checks if there exists a loop. If a loop exists, it must drop the packet; otherwise, go to step b).
b) If the received packet is originating by the receiver, it must add IP header; otherwise, go to step c).
c) If the packet is duplicated, the receiver drops it; otherwise, go to step d).
d) The receiver records the packet, pass copy of packet up to the application layer and rebroadcast the original packet.

5.3 SMF

By extending efficient flooding concept to the data forwarding plane, SMF provides an appropriate multicast forwarding capability through IP multicast packets. In our implementation, we follow the two typical SMF technical features [Appendix B]:

1) Duplicate packet detection (DPD) mechanisms for forwarding IP multicast packets.
2) Efficient relay sets mechanisms for reducing contention and congestion in wireless multi-hop scenario.

For the DPD mechanism, we adapt the header content identification-based (I-DPD) [Appendix B] to detect the duplicated packets. The detection method is given as follows:

a) If the current node is selected by the previous-hop transmitter as a relay, the packet identifier is checked against the DPD state for each possible outbound node. Otherwise, go to step d).
b) If the packet is not duplicated for an outbound node, the packet is forwarded via that node and a DPD entry is made for the given packet identifier for the node.
c) If the packet is a duplicate, drop the packet.
d) If the previous-hop transmitter is a 1-hop symmetric neighbor, A DPD entry is made for that packet, but the packet is not forwarded.
e) Otherwise, drop the packet.

As discussed in the Appendix B, SMF has four distributed methods of dynamically electing reduced relay sets that attempt to optimize the flooding of routing control messages in MANET routing peers, Source-specific Multipoint Relay (S-MPR), Non-source-specific MPR (NS-MPR), Essential Connecting Dominating Set (E-CDS), and MPR-based CDS (MPR-CDS). [77] presented the design of a SMF prototype and some initial performance results using the NRL mobile network emulation system and various optional flooding approaches within the design framework. Through performance comparison, S-MPR has relatively higher throughput and less number of forwarders, which makes it a more efficient relay set selection algorithm. Thus, we decide to implement SMF with S-MPR selection algorithm, which is described as follows:

a) For any node, say node *N*, initialize its MPR set to empty.
b) Initialize the set *N1* to include all 1-hop neighbors of *N*.
c) Initialize the set *N2* to include all 2-hop neighbors, excluding *N* and any node in *N1*.
d) Select the node *M* in *N1* with a router priority value of “Always”, add it to the MPR set, and remove all *M*’s 1-hop neighbors from *N2*. If some of *M*’s 1-hop neighbors are also the 1-hop neighbors of the nodes in *N1*, remove them from the 1-hop neighbors of those nodes in *N1*.
e) For each node *M* in *N2* which has only one 1-hop neighbor in *N1*, select it as a MPR, add it to the MPR set, and remove all *M*’s 1-hop neighbors from *N2*. If some of *M*’s 1-hop neighbors are also the 1-hop neighbors of the nodes in *N1*, remove them from the 1-hop neighbors of those nodes in *N1*.
f) If *N2* is not empty, select nodes in *N1* with the largest router priority, select it as a MPR, add it to the MPR set, and remove all *M*’s 1-hop neighbors from *N2*. If some of *M*’s 1-hop neighbors are also the 1-hop neighbors of the nodes in *N1*, remove them from the 1-hop neighbors of those nodes in *N1*.
g) Algorithm ends until *N2* is empty.

5.4 CodeBCast

CodeBCast applied network coding to a deterministic broadcast approach, resulting in a significant reduction in the number of transmissions in the network. The pseudo-code is shown as follows:

```
Upon receiving a new packet $p$ or on time-out of a buffered packet $p$
1. Update NbrRecvTable($p$);
2. if $u \notin Fwder(p)$ return;
3. if allNbrRecv($p$) return;
4. if Native($p$), then
5.     $B$ = getCodeSet();
6.     if ($|B| > 1$) then
7.         sendCodedPkts($B$);
8.     else
9.         if(!Timeout($p$)) then
```

```
10.         Queue(p, $\tau$);
11.       else sendNative(p);
12.     end if
13. end if
14. else
15. for each $q$ = decode(p)
16.       Process(q)
17.     end for
18. end if
```

Nodes exchanged their neighbors' information so that each node knew the network topology within its 2-hop neighbors. When any packet is received, the node first updates its neighbor table (line 1). For each native packet, only a subset of neighbors is delegated as forwarders; other nodes do not rebroadcast (line 2). When node *u* receives a packet from its neighbor, if all its neighbors have already received the packet based on its neighbor table, it will not rebroadcast (line 3). Otherwise, node *u* tries to see if it can get any coding opportunities by encoding the packets received and then forwards (line 5). If yes, one or more encoded packets are generated to be transmitted (line 7). If not, for delay tolerant applications, the node will buffer the packet for a random amount of time τ (line 10) and then process it. For non-delay tolerant applications, a packet is sent immediately (line 11). Finally, if the received packet is a coded packet, we decode it before processing the packet (line 15and line16).

Since simulation results in [9] showed that the Reed-Solomon based coding algorithm [Appendix C] outperformed the XOR-based algorithm [Appendix C] in terms of the coding gain and the packet delivery ratio, we will implement Reed-Solomon based coding algorithm. The pseudo-code of this algorithm is given as follows:

```
getCodeSet()
1. Pick native packet set $P$ in the output queue
2. $k$ = maxMissingPackets(N(u), P)
3. Construct encoded packet set $Q = \theta \times P$
```

```
4. Add packet ID of each packet $p \in P$ to $q_i$
5. Add the row index $i$ of $\theta$ to $q_i$
6. return $Q$
```

The algorithm constructs coded packet set $Q = \theta \times P$ (line 1-3), where P is the native packet set, θ is the $k*n$ Vandermonde matrix where 1, 2, …, n are labels of elements in the finite field. It then adds the set of native packet IDs to each coded packet (line 4). In practice, this set of IDs can be spread across the k packets. It then adds the index number of codes used (line 5).

6 PERFORMANCE EVALUATION

In this section, we evaluate the performance of Classical Flooding, SMF, CodeBCast and EBOLSR protocols, in terms of the packet delivery ratio (PDR), average end-to-end delay, energy consumption per packet and network lifetime. Moreover, the average number of EMPRs per node in the EBOLSR protocol and the average number of MPRs per node in SMF is compared as well.

All protocols are all implemented in NS2.29 [11]. Implementing Classical Flooding is relatively simple and only requires duplicate packet detection. Each packet is uniquely identified by the address of the sender and a sender-assigned ID. The version of SMF we used is the version provided by NRL [77]. This version is only supported for NS2.29, and all protocols are implemented in that NS2 version. The CodeBCast protocol has already been implemented in [78]. RS coding algorithm is used on top of the SMF broadcasting. EBOLSR is implemented as a modification of SMF.

6.1 Network Scenarios and Performance Metrics

To evaluate these four protocols, we simulated two distinct types of networks:

- WiFi Scenario
- Rescue Scenario with a more realistic radio link model based on Ricean fading (called RiceanRescue) and low-bandwidth tactical radios.

The WiFi scenario is a scenario commonly used in MANET protocol evaluations. The RiceanRescue scenario is a more realistic tactical scenario with parameters based on field data from a CRC [78] measurement study. Considering the different type of radio and bandwidth supported by WiFi and RiceanRescue scenarios, we decide to simulate the four protocols and evaluate their performance in these two scenarios. More description of the two scenarios is given as follows.

6.1.1 WiFi Scenario

25 nodes are randomly placed in a 700 meter by 700 meter area. Each node has a wireless interface that is based on IEEE 802.11 (Distributed Coordination Function, 250m transmission range, 550m sensing range, 2 Mbps data rate). The nodes move randomly

according to the Random Waypoint Mobility model [79], with different maximum speeds and 1 second pause time. More simulation parameters are shown in Table 1. For the SMF, CodeBCast and EBOLSR protocols, periodic Hello packets are transmitted by each node every 1 second and the Hello jitter is 0.1 second. A Hello jitter is a jitter interval to randomize the Hello transmission time, this helps to avoid protocol synchronization issues and packet collision. All nodes in the network have the same transmission, reception, idle and sleeping power. The initial energy of 25 nodes is uniformly distributed in [100J-800J].

Table 1 Simulation parameters for WiFi Scenario

Simulation Parameters	**Value**
Number of Nodes	25
Network Size	700m×700m
Simulation Time	600s
Propagation Model	Two-ray Ground
Medium Access Control	IEEE 802.11
Transmission Range	250m
Transmission Power	1.4W
Receiving Power	1.0W
Idle Power	0.5W
Bandwidth	2Mb/s
Packet Type	CBR
Packet Size	512bytes
Packet rate	4Pkts/s
Mobility Model	Random Way-Point
Maximum Speed	1m/s, 5m/s, 10m/s, 15m/s, 20m/s
Number of Senders	1
Number of Receivers	25

6.1.2 RiceanRescue Scenario

In the RiceanRescue scenario, 50 nodes are placed in a 40 km by 40 km area. The MAC protocol is TMDA, and the MAC bandwidth is lowered to a data rate of 128 kbps. It uses a PHY model that combines Ricean Fading and Shadowing, with parameters based on field data from a CRC [78] measurement study. The parameters are chosen such that a link has near perfect reception up to 20 km, with packet loss increasing thereafter. However, even for a transmission range of 40 km, a nontrivial amount of packets are still received (approximately 50%, given the Ricean K factor and path loss exponent we use). As the radio range increases, the network topology changes less often, so we lowered the

Hello interval to 10 seconds. Traffic is based on the sender generating a VoIP data stream with 80 byte CBR packets and a data rate of 2.4kbps. In this scenario, all nodes have the same initial energy, 1000J. More simulation parameters are shown in Table 2.

Table 2 Simulation parameters for RiceanRescue Scenario

Number of Node	50
Network Size	40000m*40000m
Simulation Time	2000s
Propagation Model	RiceanShadowing
Ricean K Factor	4.8dB
Reference Distance	d0=100m
Path Loss Exponent	3.75
Shadowing Loss Variance	4.5
Transmit Power	0.281838W
Frequency	7.75e+07
Receiving threshold	6.61502e-16
Medium Access Control	TDMA
Initial Energy	1000J
Transmission Power	1.4W
Receiving Power	1.0W
Idle Power	0.5W
Packet Type	CBR
Bandwidth	128Kb
Packet Size	80bytes
Packet rate	2.4Kbps
Number of Senders	1
Number of Receivers	50

The nodes all move with the velocity depends on the group a node belongs to. Of the 50 nodes, 3 nodes, representing command-and-control centers, are nearly stationary. 7 nodes move individually around the whole simulation area based on the Random Waypoint mobility model, with speed randomly selected between 30 and 70 km/h and 0 pause time. The remaining 40 nodes are grouped into 4 sets of ten nodes each, moving as a group. Each group of 10 nodes moves according to the Reference Point Group Mobility Model, where the reference point moves with a speed randomly selected between 30 and 70 km/h and 0 pause time. Within each group, nodes can deviate from the reference point by +/- 1 km in each direction. In addition, each of the four groups is assigned to work in one quadrant of the simulation area, with the quadrants slightly overlapping. That is, group one works in the quadrant bounded by (0, 0) and (22 km, 22 km), group 2 works in a quadrant bounded by (0, 18 km) and (18 km, 40 km), group three is within the quadrant bounded by (18 km, 0) and (40 km, 22 km), and finally group 4 operates within the quadrant bounded by (18km, 18km) and (40km, 40km). In all scenarios, the network is and remains fully connected.

6.1.3 Performance Metrics

To evaluate the protocols, the following metrics are used:

a) Packet Delivery Ratio: It is calculated as the ratio between the number of data packets received and the number of data packets which are expected to be received (the data packets sent times the number of nodes in the network).

b) Average End-to-end Delay: It is the data packet delay averaged among all delivered packets.

c) Energy Consumption per Packet: It is calculated as the ratio between the total energy consumed and the number of data packets delivered.

d) Network Lifetime:

- If at least one node fails during the simulation time, the network lifetime is defined as the time to the first node failure due to battery exhaustion. This metric is used in the WiFi scenario, where some of the nodes die before the simulation ends due to the limited energy.
- Otherwise, we evaluate the minimum remaining energy among all nodes as the simulation ends. This metric is used in the RiceanRescue scenario, where all nodes have enough energy and none will die before the simulation ends. Then we focus on the node with the minimum remaining energy in the four protocols.

We evaluate the performance in a different way in the RiceanRescue scenario. Actually, the minimum remaining energy can also show the performance in terms of energy consumption of four protocols.

e) Number of (E)MPRs per node: The average number of EMPRs (forwarders) per node in EBOLSR protocol and the average number of MPRs per node in SMF. They are both the total number selected per node over the whole simulation. The less number of forwarders, the more efficient the protocol is, because fewer transmissions provide less transmission energy consumption and avoid the network congestion. Thus, this metric can also explain why EBOLSR does not outperform SMF in terms of energy consumption per packet and the network lifetime.

6.2 Performance Comparison

In the WiFi scenario, for each protocol, four samples are collected in each simulation run, which are the PDR, average end-to-end delay, energy consumption per packet and the network lifetime. For EBOLSR and SMF, the average number of (E)MPR per node is also collected in each simulation run. Since we want to evaluate each metric as the function of the maximum node speed in this scenario, we choose 10 different node mobilities at each maximum node speed (in each mobility, all nodes move at the same maximum speed) and thus run 10 times for each metric at each maximum node speed. Finally, we average the values over 10 simulation runs at each maximum node speed and plot the figures for each metric. In the RiceanRescue scenario, we evaluate the minimum remaining energy instead of the time to the first node failure. Additionally, we will not evaluate the metrics as the function of the node mobility. For each metric, we have 10 different node mobilities (we do not care about the maximum node speed for each mobility) and get the average value for each metric over 10 simulation runs. The simulation time for both scenarios has already been given in Table 1 and Table 2.

6.2.1 WiFi Scenario

Figure 3 shows the PDR as the function of mobility for the four protocols. For all protocols, increasing the mobility of nodes leads to a drop in PDR except for Classical Flooding. Classical Flooding has the highest PDR in all mobility modes among all protocols due to the redundant nature of this protocol. Since some nodes fail during the simulation, Classical Flooding cannot achieve 100% PDR. CodeBCast has the second highest PDR. SMF and EBOLSR have almost the same results.

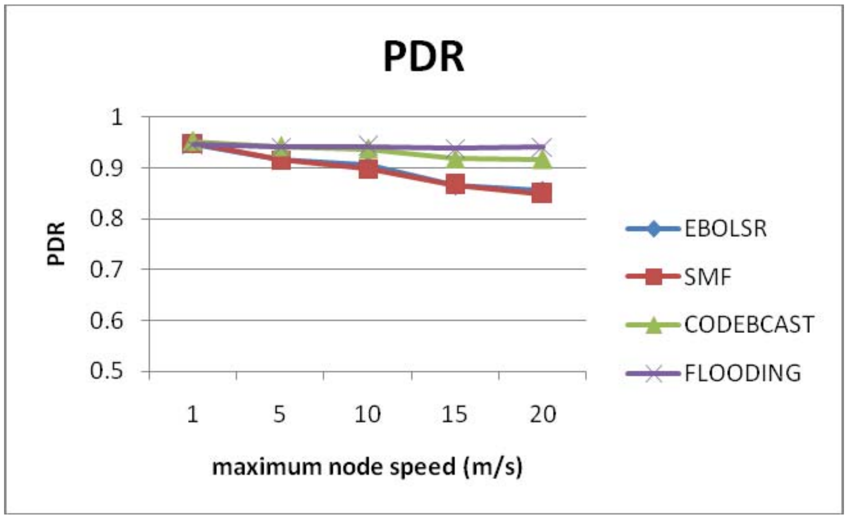


Fig. 3 PDR

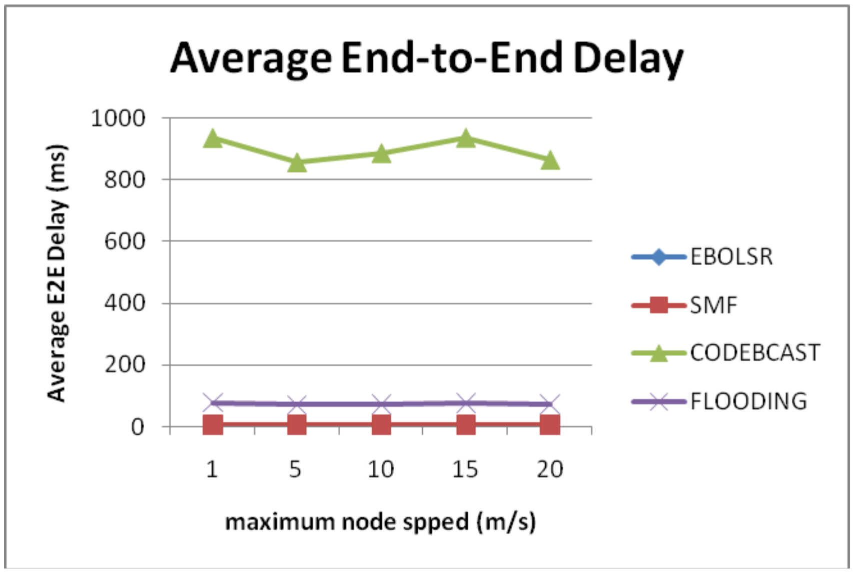


Fig. 4 Average End-to End Delay

Figure 4 shows the average end-to-end delay as the function of the mobility speed. Since all nodes in Classical Flooding rebroadcast the received packet if it is not duplicated, which can result in the network congestion, Classical Flooding has longer delay than SMF and EBOLSR. SMF and EBOLSR have the lowest delay, which are almost overlapped in the figure, while CodeBCast has the highest delay. That is because some packets need to be buffered in the sending nodes to combine with other useful packets in order for the receivers to decode. In additional, we observe that the average end-to-end delay of CodeBCast is neither an increasing function nor a decreasing function of the mobility. That is, the mobility does not influence the delay to some extent.

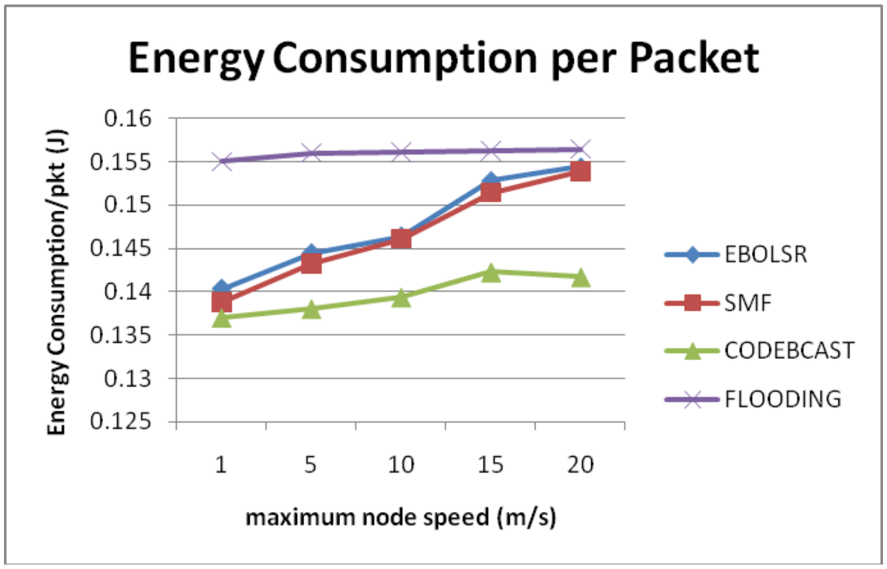


Fig. 5 Energy Consumption per Packet

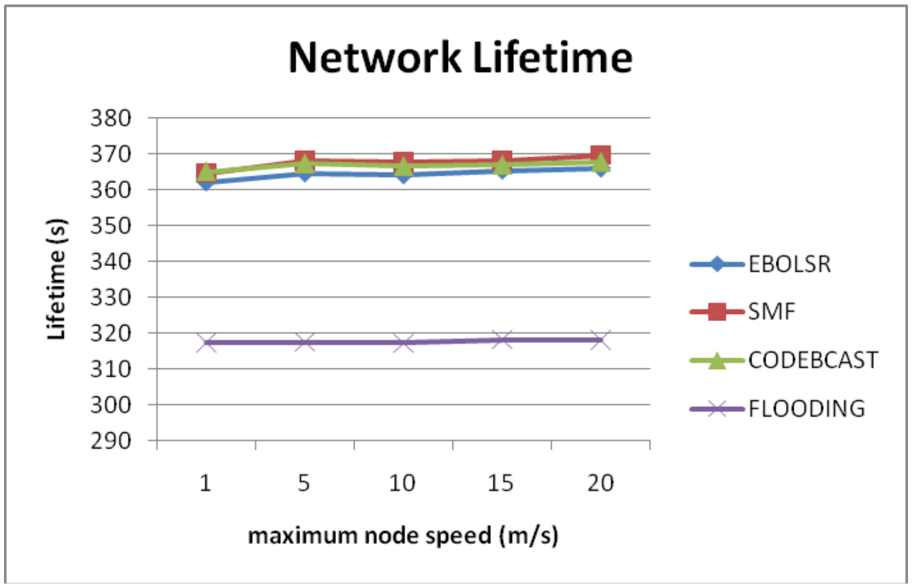


Fig. 6 Network Lifetime

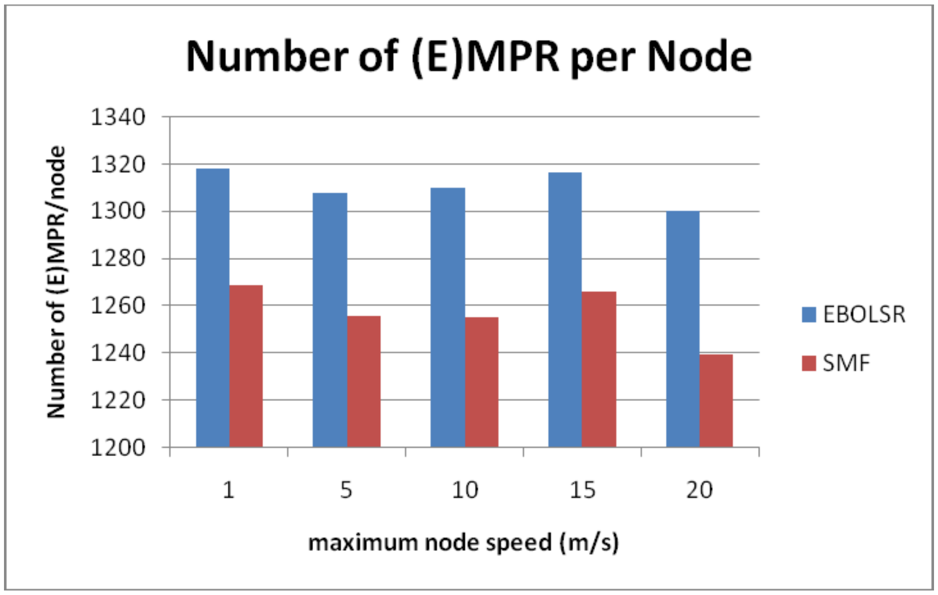


Fig. 7 Average number of (E)MPRs per node

Figure 5 shows the energy consumption per packet as a function of mobility. Among all protocols, Classical Flooding consumes the most energy due to its flooding nature, while CodeBCast consumes the least energy. SMF and EBOLSR perform in between. We also observe that the energy consumption per packet of both EBOLSR and SMF is an increasing function of mobility, while that of CodeBCast is an increasing function of mobility except at the maximum node speed 20m/s. Since we have only 10 simulation runs at each maximum node speed, inadequate statistics may lead to bumps or kinks in the performance diagram. Considering there was no work studying about the performance evaluation of CodeBCast in terms of the energy consumption per packet, we will study it in the future work through more simulation runs and statistical analysis.

Figure 6 shows the network lifetime as the function of mobility speed. Among all the protocols, Classical Flooding has the shortest lifetime since more nodes in the network rebroadcast, regardless of how their residual energy. The network lifetimes of SMF, CodeBCast and EBOLSR are similar, and it is not influenced by the mobility. From Figure 5, we notice that the energy consumption per packet of EBOLSR and SMF is higher than CodeBCast, while the network lifetime of the three protocols is similar. That is because CodeBCast has higher PDR than the EBOLSR and SMF, and we define the energy consumption per packet as the total energy consumed divided by the number of packets successfully delivered. In fact, the total energy consumption and the energy consumption of each node of these three protocols is almost the same. As a result, the network lifetime of them is also similar.

Figure 7 shows the average number of EMPRs per node in EBOLSR protocol and the average number of MPRs per node in SMF within the whole simulation time, 600s. In both protocols, each node selects its (E)MPR set when sending a Hello Packet. Since the Hello interval is the same, 1 second, in both protocols, the total number of times a node selects (E)MPRs are the same as well. From the simulation results, we observe that in the EBOLSR protocol, each node selects more relaying nodes from its 1-hop neighbors to rebroadcast than that of SMF for all mobilities. If we divide these values by the selection times and average them, each node selects about 2.2 EMPRs each time in the EBOLSR protocol and selects about 2.1 MPRs in SMF each time on average. Since all the EMPR nodes selected will participate in the rebroadcasting, this result can explain why the EBOLSR protocol does not outperform SMF in terms of energy consumption per packet and the network lifetime. Although it takes into account the residual energy of the candidate relaying nodes, still more nodes are selected to rebroadcast, which results in a higher overall energy consumption.

6.2.2 RiceanRescue Scenario

In the WiFi scenario, we have 5 different mobility models with 5 different maximum node speeds, which are 1m/s, 5m/s, 10m/s, 15m/s and 20m/s respectively, and for each mobility model with a certain maximum node speed, we have 10 scenarios. Since we evaluate each metric as the function of the maximum node speed, we average the results over 10 simulation runs at each maximum node speed. In the RiceanRescue scenario, we do not consider the maximum node speed of each scenario and have 10 simulation runs. For each simulation run, we collect the values for each metric. Finally, each metric has only one data point which is the average of the results of 10 simulations. The performance of each protocol in the RiceanRescue scenario is given in Table 3.

To evaluate the energy consumption of four protocols in a different way, each node has enough initial energy in this scenario, and none will die before the simulation ends. We therefore focus on the node with the lowest remaining energy in the network instead of evaluating the network lifetime. From the comparison results we observe that Classical Flooding provides the highest PDR, but has the highest energy consumption per packet and the lowest remaining energy among the four protocols. CodeBCast outperforms the others in terms of the energy consumption per packet, but with the longest end-to-end delay. Compared with Classical Flooding and CodeBCast, SMF and EBOLSR have relatively low PDR, but the shortest end-to-end delay. We observe that the PDR of SMF and CodeBCast is pretty much close to each other, which is different from that in the WiFi scenario, because the radio range is much longer in the RiceanRescue scenario, resulting in larger coverage area. The energy consumption per packet is similar to that of CodeBCast, but a little bit higher. The lowest remaining energy of EBOLSR is lower than SMF and CodeBCast, and SMF outperforms all other protocols, achieving the best value for the lowest remaining energy. The simulation result also shows that, during 2000 seconds simulation time, the average number of EMPRs per node in EBOLSR is only 14.38 and the average number of MPRs per node of SMF is only 13.86. The values of this metric are much lower than that of the WiFi scenario. If we divide these value by the number of times (E)MPR are selected, 200 (Hello interval is set at 10 seconds and simulation time is 2000 seconds), each node selects 0.07 (E)MPRs each time on average. That means nodes have 0 (E)MPRs most of the time. The reason is that, different from the default transmission range, 250 meters, in the WiFi scenario, the transmission range in this tactical scenario is a variable and also much longer. All other nodes are 1-hop neighbors of the selector most of the time, which result in 0 (E)MPR. In fact, each selector has 49 1-hop neighbors most of the time and 46 to 48 1-hop neighbors occasionally. Furthermore, by comparison, we observe that the PDR, energy consumption per packet and the lowest remaining energy of CodeBCast and SMF are quiet close to each other. The confidence intervals and statistically significant differences of these metrics at 95% confidence level are given in Table 4. Since these data points are the averages over 10 simulation runs of different mobility, and the PDR and the lowest remaining energy of the two protocols have overlapping confidence intervals, the differences of the PDR and the lowest remaining energy of the two protocols are not statistically significant.

Table 3 Performance Comparison in RiceanRescue Scenario

	PDR	Average E2E Delay (s)	Energy Consumption/pkt (J)	Lowest remaining Energy (J)	Number of (E)MPRs/node
Flooding	99.99%	0.1057	0.0761	451.7212	
CodeBCast	97.62%	0.5783	0.0198	854.2188	
SMF	97.23%	0.0563	0.0206	855.8510	13.86
EBOLSR	95.90%	0.0535	0.0226	844.2214	14.38

Table 4 Confidence Interval and Statistically Significant Differences
(at 95% confidence level)

	CodeBCast	SMF	Differences Statistically Significant?
PDR	(96.81%, 98.40%)	(95.67%, 98.79%)	YES
Energy Consumption/pkt	(0.0194, 0.0201)	(0.0203, 0.0209)	NO
Lowest Energy	(852.9478, 855.4898)	(854.5336, 857.1684)	YES

7 CONCLUSIONS

MANETs are faced with the problem of energy efficiency in order to maximize the network lifetime. The goal of this project is to explore energy-efficient protocols in broadcasting scenarios and compare a suitable protocol with three other broadcast protocols in MANETs. We adopted the multipoint relay selection strategy based on residual energy in the EOLSR protocol and use it in the broadcasting scenarios. This EMPR selection strategy takes into account the energy dissipated in transmission and reception up to 1-hop from the transmitter and was verified to prolong the network lifetime and increase the packet delivery rate when combined with the proposed unicast routing strategy. The proposed EBOLSR protocol is implemented as the combination of the EMPR selection strategy and the SMF broadcasting method.

In order to evaluate and compare the selected broadcast protocols, we have evaluated the performance of EBOLSR, SMF, CodeBCast and Classical Flooding in two scenarios: WiFi scenario and RiceanRescue scenario. Simulation results in both scenarios show that Classical Flooding has the highest PDR but also the highest energy consumption, while CodeBCast has the lowest energy consumption but the longest end-to-end delay. EBOLSR and SMF achieve similar results in terms of PDR, average end-to-end delay, energy consumption and lifetime, but EBOLSR does not outperform SMF with regards to energy efficiency, contrary to what we expected. To explain these results, there are several valid reasons as follows:

- Packet Size: In order to record and update the residual energy of each node's 1-hop and 2-hop neighbors, we add more fields in the Hello message to record the residual energy of the sending node's 1-hop neighbors. Each field recording the residual energy is 4 bytes and the number of fields is the number of 1-hop neighbors of the sending nodes. Since the packet size is bigger than that in SMF, it results in more energy consumption in the transmission and reception of Hello messages.

- Number of Relaying Nodes: Although EBOLSR takes neighbors' residual energy into account when selecting the relying nodes, the number of EMPRs selected is higher than the number of MPRs in SMF. In [10], the author verified that the number of EMPRs per node selected in EOLSR protocol is higher than the number of MPRs per node selected in the native OLSR protocol. More broadcasting nodes will results in more energy consumption.

- Broadcasting Network: The EOLSR protocol achieves longer network lifetime than the OLSR protocol, while our protocol does not outperform SMF in this respect. The reason is that EOLSR is a unicast routing protocol but EBOLSR is a broadcast routing protocol. After each node selects it EMPR set, the EOLSR protocol only picks one of these relaying nodes to forward the data packet, following its routing strategy to deliver the data packet from the source to the destination. While in EBOLSR protocol, all selected EMPRs must rebroadcast the data packets, which cause more energy consumption if more nodes are selected as EMPRs.

Through the performance comparison of four protocols, we conclude that CodeBCast is a good choice to broadcast data packets in networks which require high throughput and low energy consumption, and SMF is suitable to work in networks that require short end-to-end delay and low energy consumption. Although SMF has higher energy consumption than CodeBCast, it has comparable PDR, less energy consumption, and the shortest end-to-end delay in both scenarios, which suggest it to have the best overall performance. The future work in the design of energy-efficient broadcast routing protocols in MANETs should try to reduce the transmission redundancy and overall network overhead, and thus achieve the minimum energy consumption and the maximum network lifetime.

REFERENCES

[1] D. Johnson, Y. Hu, D. Maltz, Dynamic Source Routing Protocol for Mobile Ad Hoc Networks for IPv4, RFC 4728
[2] C. Perkins, E. Belding-Royer, S. Das, Ad hoc On-demand Distance Vector (AODV) Routing, RFC 3561
[3] C. E. Perkins, P. Bhagwat, Highly Dynamic Destination-Sequenced Distance Vector (DSDV) for Mobile Computers, Proc. of the Special Interest Group on Data Communication, Volume 24 , Issue 4, Oct. 1994, pp. 234-244.
[4] P. Jacquet, T. Clausen, Optimized Link State Routing Protocol (OLSR), RFC 3626
[5] S. Lee, M. Gerla, C. Chiang, On-Demand Multicast Routing Protocol, Proc. of IEEE Wireless Communications and Networking Conference, Sep. 1999, pp. 1298-1304.
[6] E. M. Royer, C. E. Perkins, Multicast Operation of the Ad-hoc On-demand Distance Vector Routing Protocol, Proc. of ACM/IEEE International Conference on Mobile Computing and Networking, Aug. 1999, pp. 207-218.
[7] E. Bommaiah, A. McAuley, R. R. Talpade, M. Liu, AMRoute: Ad hoc Multicast Routing Protocol, Internet-Draft, draft-talpade-manetamroute-00.txt, Work in progress, Aug. 1998.
[8] Network Working Group, SMF Design Team (IETF MANET WG), Simplified Multicast Forwarding for MANET (draft-ietf-manet-smf-07), Feb. 2008.
[9] L. Li, R. Ramjee, M. Buddhikot, S. Miller, Network Coding-Based Broadcast in Mobile Ad hoc Networks, 26th IEEE International Conference on Computer Communications, IEEE, May 2007, pp. 1739-1747.
[10] S. Mahfoudh, P. Minet, EOLSR: An Energy Efficient Routing Based on OLSR in Wireless Ad Hoc and Sensor Networks, Proc. of the 22nd International Conference on Advanced Information Networking and Applications, Mar. 2008, pp. 1253-1259.
[11] http://www.isi.edu/nsnam/ns/
[12] J.G. Jetcheva, D. B. Johnson, Adaptive demand-driven multicast routing in multi-hop wireless ad hoc networks, Proc. of the ACM International Symposium on Mobile Ad Hoc Networking and Computing, Oct. 2001, pp. 33-44.
[13] C. Wu, Y. C. Tay, C. K. Toh, Ad hoc Multicast Routing protocol utilizing Increasing id-numberS (AMRIS) Functional Specification, Internet-Draft, draft-ietf-manet-amris-spec-00.txt, Work in progress, Nov. 1998.
[14] J. J. Garcia-Luna-Aceves, E. L. Madruqa, The Core-Assisted Mesh Protocol, IEEE Journal on Selected Areas in Communications, Volume 17, Issue 8, Aug. 1999, pp. 1380-1394.
[15] Z. Haas, A New Routing Protocol for Reconfigurable Wireless Networks, Universal Personal Communications Record, Oct. 1997, pp. 562-566.
[16] Y. B. Ko, N. H. Vaidya, Location-aided Routing (LAR) in Mobile Ad hoc Networks, Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking, Jul. 1998, pp. 66-75.
[17] B. Williams, T. Camp, Comparison of Broadcasting Techniques for Mobile Ad hoc Networks, Proc. of the ACM Symposium on Mobile Ad Hoc Networking and Computing, 2002, pp. 194-205.
[18] Y. C. Tseng, S. Y. Ni, Y. S. Chen, and J. P. Sheu, The broadcast storm problem in a mobile ad hoc network, Wireless Networks, Volume 8, Issue 2-3, 2002, pp. 153-167.
[19] H. Lim and C. Kim, Multicast tree construction and flooding in wireless ad hoc

networks, Proc. of the ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2000, pp. 61-68.
[20] A. Qayyum, L. Viennot, and A. Laouiti, Multipoint relaying: An efficient technique for flooding in mobile wireless networks, Technical Report 3898, Institut National de Recherche en Informatique et Automatique, Mar. 2000.
[21] W. Peng and X. Lu, On the reduction of broadcast redundancy in mobile ad hoc networks, Proc. of the 1st ACM international symposium on Mobile ad hoc networking and computing, Nov. 2000, pp. 129-130.
[22] W. Peng and X. Lu, AHBP: An efficient broadcast protocol for mobile ad hoc networks. Journal of Science and Technology, Volume 16, Issue 2, Mar. 2001, pp. 114-125.
[23] W. Peng and X. Lu, Efficient broadcast in mobile ad hoc networks using connected dominating sets, Journal of Software, Volume 12, Issue 4, Apr. 2001, pp. 529-536.
[24] J. Sucec and I. Marsic, An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks, Technical Report 248, Rutgers University, Sep. 2000.
[25] P. Jacquet, V. Laouiti, P. Minet, and L. Viennot, Performance of multipoint relaying in ad hoc mobile routing protocols, Networking, 2002, pp. 387-398.
[26] J. P. Macker, J. Dean, W. Chao, Simplified Multicast Forwarding in Mobile Ad Hoc Network, Proc. of IEEE Military Communications Conference, Volume 2, Nov. 2004, pp. 744-750.
[27] S. Guo, O. Yang, Energy-aware multicasting in wireless ad hoc networks: A survey and discussion, Computer Communications, Volume 30, Issue 9, Jul. 2007, pp. 2129-2148.
[28] W. Liang, Constructing minimum-energy broadcast trees in wireless ad hoc networks, Proc. of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing, Jun. 2002, pp. 112-122.
[29] M. Cagalj, J. Hubaux, C. Enz, Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues, Proc. of the Annual International Conference on Mobile Computing and Networking, 2002, pp. 172-182.
[30] J. E. Wieselthier, G.. D. Nguyen, A. Ephremides, On the construction of energy efficient broadcast and multicast trees in wireless networks, Proc. of IEEE INFOCOM, 2000, pp. 585-594.
[31] J. E. Wieselthier, G.. D. Nguyen, A. Ephremides, Algorithms for energy-efficient multicasting in static ad hoc wireless networks, Mobile Networks and Applications, Volume 6, Issue 3, Jun. 2001, pp. 251-263.
[32] J. E. Wieselthier, G.. D. Nguyen, A. Ephremides, Energy-efficient broadcast and multicast trees in wireless networks, Mobile Networks and Applications, Volume 7, Issue 6, 2002, pp. 481-492.
[33] S. Guo and Oliver Yang, Localized Operations for Distributed Minimum Energy Multicast Algorithm in Mobile Ad Hoc Networks, IEEE Trans. Parallel and Distributed Systems, Feb. 2007, pp. 186-198.
[34] S. Guo and Oliver Yang, A Constraint Formulation for Minimum-Energy Multicast Routing in Wireless Multi-hop Networks, WINET (Wireless Network), Vol. 12.1, Feb. 2006, pp.23-32.
[35] S. Guo and Oliver Yang, QoS-Aware Minimum Energy Multicast Tree Construction in Wireless Ad Hoc Networks, Elsevier Ad Hoc Networks Journal, vol. 2/3, Apr.

2004, pp. 217 - 229.
[36] J. Cartigny, D. Simplot, I. Stojmenovic, Localized minimum-energy broadcasting in ad-hoc networks, Proc. of IEEE INFOCOM, Apr. 2003, pp. 2210-2217
[37] N. Li, J. C. Hou, L. Sha, Design and analysis of an MSTbased topology control algorithm, Proc. of IEEE INFOCOM, Apr. 2003, pp. 1702-1712.
[38] X. Li, Y. Wang, W. Song, P. Wan, O. Frieder, Localized minimum spanning tree and its applications in wireless ad hoc networks, Proc. of IEEE INFOCOM, 2004, pp. 431-442.
[39] F. Li, I. Nikolaidis, On minimum-energy broadcasting in all-wireless networks, IEEE Conference on Local Computer Networks, Nov. 2001, pp. 14-16.
[40] S. Guo, O. Yang, A dynamic multicast tree reconstruction algorithm for minimum-energy multicasting in wireless ad hoc networks, Proc. of the 2004 IEEE International Performance, Computing, and Communications Conference, Apr. 2004, pp. 637-642.
[41] F.K. Hwang, D.S. Richards, P. Winter, The Steiner Tree Problem, Amsterdam, North-Holland, The Netherlands, 1992.
[42] P. Wan, G. Calinescu, X. Li, O. Frieder, Minimum-energy multicast routing in static ad hoc wireless networks, IEEE Vehicular Technology Conference, Volume 60, Issue 6, 2004, pp. 3989-3993.
[43] B. Wang, S. K. S. Gupta, S-REMiT: a distributed algorithm for source-based energy efficient multicasting in wireless ad hoc networks, IEEE Global Telecommunications Conference, Volume 6, Dec. 2003, pp. 3519-3524.
[44] J.E. Wieselthier, G.D. Nguyen, Energy-limited wireless networking with directional antennas: the case of session-based multicasting, IEEE Conference on Computer Communications, 2002, pp. 190-199.
[45] S. Guo, O. Yang, Minimum-energy multicast in wireless ad hoc networks with adaptive antennas: MILP formulations and heuristic algorithms, IEEE Transaction on Mobile Computing, Volume 5, Issue 4, Apr. 2006, pp. 333-346.
[46] S. Guo, O. Yang, Joint optimization of energy consumption and antenna orientation for multicasting in static ad hoc wireless networks, IEEE Transaction on Wireless Communications, Volume 5, Issue 9, Sep. 2006, pp. 2563-2568.
[47] A. K. Das, R. J. Marks, M. A. El-Sharkawi, P. Arabshahi, A. Gray, MDLT: a polynomial time optimal algorithm for maximization of time-to-first-failure in energy-constrained broadcast wireless networks, IEEE Global Telecommunications Conference, Dec. 2003, pp. 362-366.
[48] I. Kang, R. Poovendran, On the lifetime extension of energy-efficient multihop broadcast networks, World Congress on Computational Intelligence, May 2002, pp. 365-370.
[49] S. Guo and Oliver Yang , Maximizing Multicast Communication Lifetime in Wireless Mobile Ad Hoc Networks, IEEE Transaction on Vehicular Technology, Vol. 57, No.4, Jul. 2008. pp. 2414-2425.
[50] S. Guo and Oliver Yang, A Framework for the Multicast Lifetime Maximization Problem in Energy-Constrained Wireless Ad-hoc Networks, in ACM WINET (Wireless Network), 2007. [electronic version: 10.1007/s11276-007-0041-x]
[51] B. Floreen, P. Kaski, J. Kohonen, P. Orponen, Multicast time maximization in energy constrained wireless networks, Proc. of the Joint Workshop on Foundations of

Mobile Computing, 2003, pp. 50-58.

[52] B. Wang, S. K. S. Gupta, On maximizing lifetime of multicast trees in wireless ad hoc networks, International Conference on Parallel Processing, Oct. 2003, pp. 333-340.

[53] S. Guo, Oliver Yang and Victor Leung, Distributed Min-Max Tree Algorithms for Maximum-Lifetime Multicast in Wireless Ad Hoc Networks with Omni-directional or Directional Antennas, EURASIP Journal on Wireless Communications and Networking, vol. 2007, Article ID 98938, 10 pages, 2007.

[54] M. X. Cheng, J. Sun, M. Min,Y. Li, W. Wu, Energy-efficient broadcast and multicast routing in ad hoc wireless networks, Proc. of the 2003 IEEE International Performance, Computing, and Communications Conference, Apr. 2003, pp. 87-94.

[55] I. Kang, R. Poovendran, Maximizing static network lifetime of wireless broadcast ad hoc networks, IEEE International Conference on Communications, Volume 3, May 2003, pp. 2256-2261.

[56] S. Guo, O. Yang, Multicast lifetime maximization for energy constrained wireless ad-hoc networks with directional antennas, IEEE Global Telecommunications Conference, Volume 6, Dec. 2004, pp. 4120-4124.

[57] L. Georgiadis, Bottleneck multicast trees in linear time, IEEE Communications Letter, Volume 7, Issue 11, Nov. 2003, pp. 564-566.

[58] S. Guo, V. Leung, O. Yang, A scalable distributed multicast algorithm for lifetime maximization in large-scale resource-limited multihop wireless networks, International Wireless Communications and Mobile Computing Conference, Jul. 2006, pp. 419-424.

[59] Y. T. Hou, Y. Shi, H. D. Sherali, J. E. Wieselthier, Online lifetime-centric multicast routing for ad hoc networks with directional antennas, Proc. of IEEE INFOCOM, 2005, pp. 761-772.

[60] S. Guo, O. Yang, Formulation of Optimal tree construction for maximum lifetime multicasting in wireless ad-hoc networks with adaptive antennas, 2005 IEEE International Conference on Communications, May 2005, pp. 3370-3374.

[61] S. Guo, V. Leung, O. Yang, Distributed multicast algorithms for lifetime maximization in wireless ad hoc networks with omni-directional and directional antennas, IEEE Global Telecommunications Conference, Nov. 2006, pp. 1-5.

[62] R. Ahlswede, C. Ning, S. Li, R. W. Yeung, Network Information Flow, IEEE Transactions on Information Theory, Volume 46, Issue 4, Jul. 2000, pp. 1204-1216.

[63] R. Ahlswede, C. Ning, S. Li, R. W. Yeung, Network Information Flow, IEEE Transactions on Information Theory, Volume 46, Issue 4, Jul. 2000, pp. 1204-1216.

[64] R. Koetter, M. Medard, An Algebraic Approach to Network Coding, IEEE Transactions on Networking, Volume 11, Issue 5, Oct. 2003, pp. 782-795.

[65] S. Li, R. W. Yeung, C. Ning, Linear Network Coding, IEEE Trans. on Information Theory, Volume 49, Issue 2, Feb. 2003, pp. 371-381.

[66] D. S. Lun, M. Medard, M. Effros, On coding for reliable communication over packet networks, 42nd Annual Allerton Conference on Communication, Control, and Computing, Sep.–Oct. 2004, invited paper.

[67] D. S. Lun, M. Medard, R. Koetter, Efficient Operation of Wireless Packet Networks Using Network Coding, Proc. of International Workshop on Convergent Technologies, Jun. 2005, invited paper.

[68] J. Yuan, Z. Li, W. Yu, B. Li, A Cross-Layer Optimization Framework for Multicast in Multi-hop Wireless Networks, Proc. of the First International Conference on Wireless Internet, Jun. 2005, pp. 47-54.

[69] F. Xie, D. Lei, B. Yong, C. Lan, Popularity Aware Scheduling for Network Coding Based Content Distribution in Ad Hoc Networks, The 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Sep. 2007, pp. 1-5.

[70] V. Bharghavan, B. Das, Routing in ad hoc networks using minimum connected dominating sets, Proc. of the International Conference on Communications, Montreal, Canada, Jun. 1997, pp. 157-165.

[71] S. Pleisch, M. Balakrishnan, K. Birman, and R. Renesse, MISTRAL: Efficient flooding in mobile ad-hoc networks, Proc. of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2006, pp. 1-12.

[72] S. Yang, J. Wu, M. Cardei, Efficient Broadcast in MANETs Using Network Coding and Directional Antennas, The 27th Conference on Computer Communications, IEEE, Apr. 2008, pp. 1499-1507.

[73] Y. Wu, S. Kung, Reduced-complexity Network Coding for Multicast over Ad Hoc Networks, Acoustics, Speech, and Signal Processing, Mar. 2005, pp. 501-504.

[74] C. Fragouli, J. Widmer, J. Y. Le Boudec, A Network Coding Approach to Energy Efficient Broadcasting: from Theory to Practice, 25th IEEE International Conference on Computer Communications, Apr. 2006, pp. 1-11.

[75] Y. Wu, P. A. Chou, S. Kung, Minimum-energy multicast in mobile ad hoc networks using network coding, IEEE Transactions on Communications, Volume 53, Issue 10, Oct. 2005, pp. 1906-1918.

[76] M. Cagalj, J.-P. Hubaux, C. Enz, Minimum-energy broadcast in all wireless networks: NP-completeness and distribution issues, Proc. of International Conference on Mobile Computing and Networks, 2002, pp.172-182.

[77] NRL SMF project, http://cs.itd.nrl.navy.mil/work/smf/index.php

[78] T. Kunz, Efficiently Supporting One-to-Many and Many-to-Many Communication Patterns in Narrowband Tactical Networks: Flooding, Efficient Broadcasting, and Network Coding, technical report, Mar. 2009.

[79] J. Broch, D. Maltz, D. Johnson, Y.C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, in Proc. of ACM/IEEE International Conference on Mobile Computing and Networking, Oct. 1998, pp. 85-97.

APPENDIX A   Code Modification of EBOLSR

To implement the EBOLSR protocol, we modified the code based on the SMF code that can be downloaded from NRL's website, http://cs.itd.nrl.navy.mil/work/smf/index.php. Some important code modifications in red color are given as follows:

<olsr_packet_types.cpp>

A.1   Hello Message Processing

```
int
HelloMessage::pack(char* buffer,int maxSize){
   int sizeused=8;
   ((UINT16*)buffer)[0]=htons(reserved1);
   ((UINT8*)buffer)[2]=htime;
   ((UINT8*)buffer)[3]=willingness;
   ((float*)buffer)[1]=residual_energy; // add fields in Hello message to record residual energy
   int messagepacksize=messages.pack(&buffer[sizeused],maxSize-sizeused);
   if(messagepacksize<0){
      fprintf(stderr,"HelloMessage::pack error packing link messages, %d is my maxSize, %d is my sizeused, %d is my willingness\n",maxSize,sizeused,willingness);
      return -1;
   }
   sizeused+=messagepacksize;
   return sizeused;
}

int
HelloMessage::unpack(char* buffer,int maxSize, ProtoAddress::Type ipvMode){
   reserved1=ntohs(((UINT16*)buffer)[0]);
   htime=(UINT8)buffer[2];
   willingness=(UINT8)buffer[3];
   residual_energy = ((float*)buffer)[1];//record residual energy of the neighbor who sent the Hello
   int totalsizeused=8;//due to a new 32-bit field to add residual energy
   int sizeused=0,maxloop=0;
   for(int i=8;i<maxSize;i+=sizeused){
      LinkMessage newmessage;
      maxloop++;
      sizeused = newmessage.unpack(&(buffer[i]),maxSize-i,ipvMode);
      totalsizeused+=sizeused;
      addLinkMessage(&newmessage);
      if(maxloop>500) {
         fprintf(stderr,"HelloMessage::unpack may have infinate loop, breaking\n");break;
      }
   }
   if(totalsizeused!=maxSize){
      fprintf(stderr,"HelloMessage::unpack:: maxSize %d should be set equal to what totalsizeuesed %d ends up being and it isn't!\n",maxSize,totalsizeused);
   }
   return totalsizeused;
}
```

A.2   Add Residual Energy in the Link Message

```
bool
LinkMessage::addNeighborEnergy(ProtoAddress *newAddress, float energy){
   bool returnvalue = false;
   reserved = 2; //this link message contains residual energy information
     if(newAddress->GetType() == ProtoAddress::IPv4) {
       UINT32 ipv4addr = newAddress->IPv4GetAddress();
       ipv4addr=htonl(ipv4addr);
       size+=sizeof(newAddress->IPv4GetAddress())+4; // important to keep current size up to date;
       size+=4;
       size+=4;
       returnvalue = neighbors.append((char*)&ipv4addr,4);
       returnvalue &= residual_energy.append((char*)&energy,4);
   } else if(newAddress->GetType() == ProtoAddress::IPv6){ //ipv6
       size+=16;
       size+=4;
       returnvalue = neighbors.append((char*)(newAddress->GetRawHostAddress()),16);
       returnvalue &= residual_energy.append((char*)&energy,4);
   } else if(newAddress->GetType() == ProtoAddress::SIM){ //sim
       if(newAddress->GetLength()>4){ //can not currently be greater than 4 bytes long
          fprintf(stderr,"Linkmessage::addNeighbor: Error because simulation address is longer than 4 bytes
long, not supported at this time!\n");
          return false;
       }
       size+=4;
       size+=4;
       UINT32 ipv4addr=0; //using uint32 to buffer space
       memcpy(&ipv4addr,newAddress->GetRawHostAddress(),newAddress->GetLength());
       returnvalue = neighbors.append((char*)&ipv4addr,4);
       returnvalue &= residual_energy.append((char*)&energy,4);
   }
     return returnvalue;
}
```

<nrlolsr.cpp>

A.3 Update Neighbors' Information

```
void
Nrlolsr::update_nbr(ProtoAddress id,int status,UINT8 spfValue, UINT8 minmaxValue, UINT8 Vtime,
       UINT8 willingness, float energy) {
   NbrTuple *tuple_pt, *children, *parents;
   int updateMprs=0;
   tuple_pt = nbr_list.FindObject(id);
   if(tuple_pt==NULL){
          }
   switch(status) {
   case ASYM_LINKv4:
           if(!id.HostIsEqual(myaddress)){ // shouldn't ever happen
          if(tuple_pt) {
                    if(tuple_pt->N_status!=SYM_LINKv4 && tuple_pt->N_status!=MPR_LINKv4){
                          //if link is asym then update the object and move it to the back keeping asym status
                          nbr_list.RemoveCurrent();
                          DMSG(8,"hello message reieved konectivity %f is
       improving\n",tuple_pt->konectivity);
                          tuple_pt->konectivity=alpha*tuple_pt->konectivity+(1-alpha);
                          if(tuple_pt->konectivity>T_up && tuple_pt->N_status==PENDING_LINK){
```

```
                tuple_pt->N_status=ASYM_LINKv4;
            }
            tuple_pt->N_time=InlineGetCurrentTime() + mantissatodouble(Vtime);
            tuple_pt->N_willingness = willingness;
            if(tuple_pt->N_residual_energy > energy)//added for EMPR selection
                tuple_pt->N_residual_energy = energy;
            nbr_list.QueueObjectEnergySort(tuple_pt);//sort by residual energy
        }
    } else {
        // check to see if its a two hop neighbor
        tuple_pt = nbr_2hop_list.FindObject(id);
        if(tuple_pt) {
            //set queue up for being one hop neighbor (handling timeouts of children)
            tuple_pt->SetHoldTime(mantissatodouble(Vtime));
            //add to 1 hop list but don't erase 2 hop list entry
            tuple_pt->N_time=InlineGetCurrentTime() + mantissatodouble(Vtime);
            tuple_pt->konectivity=(T_up+T_down)/2;
            DMSG(8,"hello message reieved from known 2 hop neighbor setting konectivity
to %f\n",tuple_pt->konectivity);
            if(tuple_pt->konectivity>T_up){
                tuple_pt->N_status=ASYM_LINKv4;
            } else {
                tuple_pt->N_status=PENDING_LINK;
            }
            tuple_pt->N_willingness = willingness;
            tuple_pt->N_residual_energy = energy;//added for EMPR selection
            nbr_list.QueueObjectEnergySort(tuple_pt);//sort by residual energy
        } else {
            tuple_pt = new NbrTuple;
            tuple_pt->SetHoldTime(mantissatodouble(Vtime)); //set the queue up for the Vtime
value
            tuple_pt->N_addr=id;
            tuple_pt->konectivity=(T_up+T_down)/2;

            DMSG(8,"hello message reieved from new neighbor setting konectivity
to %f\n",tuple_pt->konectivity);
            if(tuple_pt->konectivity>T_up){
                tuple_pt->N_status=ASYM_LINKv4;
                tuple_pt->hop=1; //ok to do this here cause its a new neighbor
            } else {
                tuple_pt->N_status=PENDING_LINK;
                tuple_pt->hop=1;    //ok to do this here cause its a new neighbor
            }
            tuple_pt->N_time=InlineGetCurrentTime() + Neighb_Hold_Time;
            tuple_pt->N_willingness = willingness;
            tuple_pt->N_residual_energy = energy;//added for EMPR selection
            nbr_list.QueueObjectEnergySort(tuple_pt);//sort by residual energy
        }
    }
}
tuple_pt->recievedHello=1; //used for historisis
if(tuple_pt->N_status==PENDING_LINK){
  return 0;
} else {
    return 1;
}
```

```
    break;
  case SYM_LINKv4:    // can be mpr link as well
    if(tuple_pt) { //should always pass if done correctly
      // update link time
      if(tuple_pt->N_status==ASYM_LINKv4){
                tuple_pt->hop=1;
                tuple_pt->N_status=SYM_LINKv4;
                //link up known children

    for(children=(tuple_pt->children).PeekInit();children!=NULL;children=(tuple_pt->children).PeekNex
    t()){
                    if(children!=NULL){
                        if((children->stepparents).FindObject(tuple_pt->N_addr)){  //remove  stepparent
    status
                            (children->stepparents).RemoveCurrent();
                        } else {
                            DMSG(0,"didn't  find  stepparents  in  SYM_LINKv4  area!!!   updated
    node %s with child ",tuple_pt->N_addr.GetHostString());
                            DMSG(0,"%s          who          doesn't          know          about
    ",children->N_addr.GetHostString());
                            DMSG(0,"%s\n",tuple_pt->N_addr.GetHostString());
                        }
                        (children->parents).QueueObject(tuple_pt); // link the child to parent node
                        if(nbr_2hop_list.FindObject(children->N_addr)){//need  to  take  out  if  its  there
    already to update time
                            nbr_2hop_list.RemoveCurrent();
                        } else {
                            // is okay cause could have been only a 1 hop neighbor
                        }
                        nbr_2hop_list.QueueObject(children); // add to two hop list
                    }
                }
                updateMprs=1;
      }
      nbr_list.RemoveCurrent();
      tuple_pt->N_time=InlineGetCurrentTime() + Neighb_Hold_Time;
      tuple_pt->N_willingness = willingness;
      tuple_pt->N_residual_energy = energy;//added for EMPR selection
      nbr_list.QueueObjectEnergySort(tuple_pt);//sort by residual energy
      //this call is made only in send hello
    }
    else {
      DMSG(0,"didn't pass for some reason nrouter/nbr_update,sym_link");}
    break;
  case LOST_LINKv4:    //recieved a lost link from a neighbor you can still hear
    if(tuple_pt) { //should always pass
      // check and remove nodes children
      nbr_list.RemoveCurrent();

    for(children=(tuple_pt->children).PeekInit();children!=NULL;children=(tuple_pt->children).PeekNext
    ()){
    if(children!=NULL){
      //abandon children
      if((children->parents).FindObject(tuple_pt->N_addr))
        (children->parents).RemoveCurrent();
      else if((children->stepparents).FindObject(tuple_pt->N_addr))
```

```
        (children->stepparents).RemoveCurrent();
    if((children->parents).IsEmpty()){
        //child lost last parent, child runs free
        fflush(stdout);
        nbr_2hop_list.FindObject(children->N_addr);
        nbr_2hop_list.RemoveCurrent();
        if(!nbr_list.FindObject(children->N_addr)){ //check to see if it was exclusivly a 2 hop neighbor
            //get rid of step children

for(parents=(children->children).PeekInit();parents!=NULL;parents=(children->children).PeekNext())
{
 if(parents!=NULL){
    if((parents->stepparents).FindObject(children->N_addr)){
        (parents->stepparents).RemoveCurrent();
    } else {
        //error statements shouldn't enter here
        if((parents->parents).FindObject(children->N_addr)){
            DMSG(0,"missing stepparents link for node %s to stepparent
",parents->N_addr.GetHostString());
            DMSG(0,"%s in LOST_LINKv4 area: /n",children->N_addr.GetHostString());
            DMSG(0,"but did find parent link! wasn't moved to stepparent correctly someplace in past!");
        } else {
            DMSG(0,"missing stepparents link for node %s to stepparent
",parents->N_addr.GetHostString());
            DMSG(0,"%s in LOST_LINKv4 area: no parent link
found",children->N_addr.GetHostString());
        }
        //end error stantments
    }
 }
            }
            (children->children).Clear();
            //get rid of step parents

for(parents=(children->stepparents).PeekInit();parents!=NULL;parents=(children->stepparents).Peek
Next()){
 if(parents!=NULL){
    if(!parents->N_addr.HostIsEqual(tuple_pt->N_addr)){
        if((parents->children).FindObject(children->N_addr)){
            (parents->children).RemoveCurrent();
        } else {
            DMSG(0,"missing childlink for node %s to child ",parents->N_addr.GetHostString());
            DMSG(0,"%s in LOST_LINKv4 area: ",children->N_addr.GetHostString());
            DMSG(0,"%s stepparent was ",children->N_addr.GetHostString());
            DMSG(0,"%s\n",parents->N_addr.GetHostString());
        }
    }
 }
            }
            (children->stepparents).Clear();
            delete children;
        }
    }
}
    }
    (tuple_pt->children).Clear();
```

```
            NbrTuple *mprtuple;
            if((mprtuple = mprSelectorList.FindObject(tuple_pt->N_addr))){
        mprSelectorList.RemoveCurrent();
        delete mprtuple;
        updateSmfForwardingInfo = true;    //send updated mpr selector list to send pipe if open

        // LP 9-16-05 - added for Opnet statistic
#ifdef OPNET
        if (mprSelectorList.IsEmpty()){
         MPR_decreased_flag = OPNET_TRUE;
         }
#endif
        // end LP
            }
            //update current node
            if((tuple_pt->parents).IsEmpty()){ // has no parents can remain a 1 hop
        tuple_pt->hop=1;
            }
            else
        tuple_pt->hop=2;                         // is now a 2 hop neighbor
            tuple_pt->N_time=InlineGetCurrentTime() + Neighb_Hold_Time;
            tuple_pt->N_status=ASYM_LINKv4;
            tuple_pt->N_willingness = willingness;
            tuple_pt->N_residual_energy = energy;//added for EMPR selection
            nbr_list.QueueObjectEnergySort(tuple_pt);//sort by residual energy
        }
    }
    return 1;
}
```

A.4 Update 2-hop Neighbor's Information

```
void
Nrlolsr::update_2hop_nbrEnergy(ProtoAddress onehop_addr,ProtoAddress twohop_addr,float
        energy){//added for EMPR selection
    NbrTuple *tuple_pt,*parent_tuple_pt,*other_tuple_pt;
    int updateMprs=0;
    int parentcheck=0;
    int errortype=0;
    parent_tuple_pt=nbr_list.FindObject(onehop_addr);
    parentcheck=parent_tuple_pt->N_status==MPR_LINKv4                                   ||
        parent_tuple_pt->N_status==SYM_LINKv4;

    if((tuple_pt=nbr_2hop_list.FindObject(onehop_addr,twohop_addr))){ //returns two hop guy
        if(tuple_pt->N_residual_energy > energy)
            tuple_pt->N_residual_energy=energy;
        if(parentcheck){
            fflush(stdout);
            //update time
            if((parent_tuple_pt=(tuple_pt->parents).FindObject(onehop_addr))){
        errortype=4;
        (tuple_pt->parents).RemoveCurrent(); // removes the pointer pointing to the parent so it can be
        updated
        (tuple_pt->parents).QueueObject(parent_tuple_pt);
        nbr_2hop_list.RemoveCurrent();
        nbr_2hop_list.QueueObject(tuple_pt);
        }
        else {
    // was a step child check to see if neighbor is true one hop
    errortype=5;
    parent_tuple_pt=(tuple_pt->stepparents).FindObject(onehop_addr);
    (tuple_pt->stepparents).RemoveCurrent(); // removes the pointer pointing to the parent so it can be
    updated
    if(parent_tuple_pt->N_status==MPR_LINKv4 || parent_tuple_pt->N_status==SYM_LINKv4) {
        updateMprs=1;
        (tuple_pt->parents).QueueObject(parent_tuple_pt);
        nbr_2hop_list.RemoveCurrent();
        nbr_2hop_list.QueueObject(tuple_pt);
    }
    else {
        (tuple_pt->stepparents).QueueObject(parent_tuple_pt);
    }
        }
        other_tuple_pt=(parent_tuple_pt->children).FindObject(twohop_addr);
        fflush(stdout);
        (parent_tuple_pt->children).RemoveCurrent(); // removes the pointer pointing to the child so it can
    be updated
        (parent_tuple_pt->children).QueueObject(other_tuple_pt);    // updating the time again
    }
}
else if((tuple_pt=nbr_2hop_list.FindObject(twohop_addr))){
    //is already someones two hop neighbor just link them together
    if(parent_tuple_pt->N_status==MPR_LINKv4      ||      parent_tuple_pt->N_status==SYM_LINKv4)
    { //make sure neighbor is true one hop
        errortype=6;
        updateMprs=1;
        (tuple_pt->parents).QueueObject(parent_tuple_pt); // link child to parent node
        nbr_2hop_list.RemoveCurrent();    // next two lines update the timeout value of the 2 hop neighbor
        nbr_2hop_list.QueueObject(tuple_pt);
    } else {
        errortype=7;
        (tuple_pt->stepparents).QueueObject(parent_tuple_pt); //link child to node which isn't a real one
    hop node yet
    }
    (parent_tuple_pt->children).QueueObject(tuple_pt); // link parent to child node
}
else if((tuple_pt=nbr_list.FindObject(twohop_addr))){
    //is already a one hop neighbor just link them together
    if(parent_tuple_pt->N_status==MPR_LINKv4      ||      parent_tuple_pt->N_status==SYM_LINKv4)
    { //make sure neighbor is true one hop
        errortype=8;
        updateMprs=1;
        (tuple_pt->parents).QueueObject(parent_tuple_pt); // link the child to parent node
        nbr_2hop_list.QueueObject(tuple_pt); // add to two hop list
        (parent_tuple_pt->children).QueueObject(tuple_pt); // link the parent to the child node
    } else {
        //do nothing not a real neighbor yet
        errortype=9;
    }
}
else {
    if(parent_tuple_pt->N_status==MPR_LINKv4      ||      parent_tuple_pt->N_status==SYM_LINKv4)
    { //make sure neighbor is true one hop
        errortype=10;
        //no existing 2 hop neighbor with given twohop_addr
        // make and link them together

        updateMprs=1;
        tuple_pt=new NbrTuple; // make new neighbor
        tuple_pt->N_addr=twohop_addr;
        tuple_pt->hop=2;
        tuple_pt->N_residual_energy=energy;
        (tuple_pt->parents).QueueObject(parent_tuple_pt);// link child to parent node
        (parent_tuple_pt->children).QueueObject(tuple_pt); // link parent to child node
        nbr_2hop_list.QueueObject(tuple_pt);    // add to two hop list
        //tuple_pt is only valid for use as a 2 hop right now the queue time is not set up properly as Vtime
    for it is not known
    }
    else{
        errortype=11;
        // may make node and add to stepparents in the future
    }

}

}//end Nrlolsr::update_2hop_nbr(onehop,twohop)
```

A.5 Select EMPRs

```
void
Nrlolsr::selectempr() {
  DMSG(6,"Enter:          Nrlolsr::selectempr         of          node          %s          with          mssn
    of %d()\n",myaddress.GetHostString(),mssn);
  printCurrentTable(10);
  int numberOfParents;
  NbrTuple *child, *parent, *newmpr=NULL;
  float p_t=1.4;
  float p_r=1.0;
  mssn++;
  mssn=mssn % MAXMSSN;
  for(parent=nbr_list.PeekInit();parent!=NULL;parent=nbr_list.PeekNext()){
    if(parent!=NULL){
      if(parent->N_status==MPR_LINKv4)
        parent->N_status=SYM_LINKv4;
      parent->tdegree=0;
      parent->cdegree=0;
      parent->energy_criteria=parent->N_residual_energy/(p_t+p_r);
      for(child=(parent->children).PeekInit();child!=NULL;child=(parent->children).PeekNext()){
    if(child!=NULL){
      if((child->hop==2)){
          child->energy_criteria=child->N_residual_energy/(2*p_r);
          if(parent->energy_criteria > child->energy_criteria && child->energy_criteria!=0 ){
             parent->energy_criteria=child->energy_criteria;
          }
      }
        }
      }
    }
  }
```

```
    for(child=nbr_2hop_list.PeekInit();child!=NULL;child=nbr_2hop_list.PeekNext()){
      if(child!=NULL){
        child->covered=0;
        if(child->hop==2){
                if(!nbr_list.FindObject(child->N_addr)){
                    numberOfParents=0;
      for(parent=(child->parents).PeekInit();parent!=NULL;parent=(child->parents).PeekNext()){
                        if(parent!=NULL){
                            if(TupleLinkIsUp(parent)){
                                numberOfParents++;
                                parent->tdegree++;
                                parent->cdegree++;
                            }
                        }
                    }
                    }
            }
                    }
    }

  //for printing out degrees
      if(olsrDebugValue>=10){
      for(parent=nbr_list.PeekInit();parent!=NULL;parent=nbr_list.PeekNext()){
           DMSG(10,"%s has degree %d\n",parent->N_addr.GetHostString(),parent->cdegree);
      }
      }

    while(!Is_all_2hop_covered()){
      float highestenergy=0;
      for(parent=nbr_list.PeekInit();parent!=NULL;parent=nbr_list.PeekNext()){
        if(parent!=NULL){
          if(parent->N_status==SYM_LINKv4){////!=LINK_DOWN
                    if(highestenergy < parent->energy_criteria){
        highestenergy = parent->energy_criteria;
        newmpr=parent;
           } else if(highestenergy == parent->energy_criteria){
        if(parent->cdegree > newmpr->cdegree)
          newmpr = parent;
           }
           }
        }
      }
      if(newmpr!=NULL && (newmpr->energy_criteria != 0) && highestenergy!=0){
        makeempr(newmpr);
      }
        }

    optimize_empr();
    int num_of_empr=0;
    int num_of_nbr=0;
    for(parent=nbr_list.PeekInit();parent!=NULL;parent=nbr_list.PeekNext()){
      if(parent!=NULL){
      num_of_nbr+=1;
      if(parent->N_status==MPR_LINKv4){
        num_of_empr+=1;
        }
        }
    }
    printf("Node                %s's              empr              nubmer              %d              and
        nbrs %d\n",myaddress.GetHostString(),num_of_empr,num_of_nbr);
}
```

A.6 Check if All 2-hop Neighbors are Covered by the Selected EMPRs

```
bool
Nrlolsr::Is_all_2hop_covered() {
    NbrTuple *child;
    if(nbr_2hop_list.IsEmpty()){
        return true;
    }else {
        for(child=nbr_2hop_list.PeekInit();child!=NULL;child=nbr_2hop_list.PeekNext()){
            if(!(bool)nbr_list.FindObject(child->N_addr) && child->N_status!=SYM_LINKv4){
                        if(child->covered == 0){
                            return false;
                        }
            }
        }
    }
    return true;
}
```

A.7 Optimize the EMPR Set

```
void
Nrlolsr::optimize_empr() {
        NbrTuple *parent, *child;
        for(parent=nbr_list.PeekInit();parent!=NULL;parent=nbr_list.PeekNext()){
            if(parent!=NULL){
                if(parent->N_status==MPR_LINKv4){////go through emprs
        bool to_be_removed=true;
        for(child=(parent->children).PeekInit();child!=NULL;child=(parent->children).PeekNext()){
            if(child!=NULL){
                if((child->hop==2)){
                if(child->covered<=1){
                    to_be_removed=false;
                    break;
                }
                }
            }
        }
        if(to_be_removed && parent!=NULL)
            remove_empr(parent);
                }
            }
        }
}
```

A.8 Remove EMPRs for Optimization

```
void
Nrlolsr::remove_empr(NbrTuple *parent) {
        NbrTuple *child;
        if(parent->N_status==MPR_LINKv4)
```

```
      parent->N_status=SYM_LINKv4;
    for(child=(parent->children).PeekInit();child!=NULL;child=(parent->children).PeekNext()){
            if(child!=NULL){
        if((child->hop==2)){
          child->covered -= 1;
        }
            }
    }
    }
```

A.9 Make an EMPR and Update the Nodal Degree of All the Other 1-hop Neighbors

```
void
Nrlolsr::makeempr(NbrTuple *parent) { // updates the current degrees with the parent node selected
  NbrTuple *child, *stepparent;
  if(parent->N_status!=MPR_LINKv4){ // check to see if its an mpr already
    parent->N_status=MPR_LINKv4;
    for(child=(parent->children).PeekInit();child!=NULL;child=(parent->children).PeekNext()){
      if(child!=NULL){
    if((child->hop==2) || (child->N_status!=SYM_LINKv4 && child->N_status!=MPR_LINKv4)){
      child->covered += 1;

    for(stepparent=(child->parents).PeekInit();stepparent!=NULL;stepparent=(child->parents).PeekNext()
    ){
        if(stepparent!=NULL){
          stepparent->cdegree--;
        }
      }
    }
      }
    }
  }
}
```

APPENDIX B SMF

B.1 Duplicate Packet Detection (DPD)

Duplicate packet detection (DPD) is a common requirement in MANET packet forwarding because packets may be transmitted from the same physical interface where they arrived and nodes may also receive copies of previously transmitted packets from other forwarding neighbors. Hence, detection and avoidance of duplicate packet forwarding must be implemented using some sort of packet identification. Usually, a history of recently processed packets is recorded in order to compare with the incoming packets. For both IPv4 and IPv6, SMF has two basic approaches to duplicate packet detection: header content identification-based (I-DPD) and hash-based (H-DPD) duplicate detection. For I-DPD, packets are identified using explicit identifiers in the IP header. The specific identifier depends on the IP protocol version and the type of packet. For example, IPv4 provides an "Identification" field that can assist the DPD mechanism, and packets which contain IPSec protocol headers also provide suitable packet identifiers. These identifier fields are unique within the context of source address, destination address, protocol type, and other header fields depending on the type of identifier used for DPD. Similarly, for H-DPD, the packet hash values will be kept with respect to at least the source address to help ensure hash collision avoidance. [8] has detailed descriptions about I-DPD and H-DPD for different types of packets.

B.2 Relay Sets Selection

Another goal of SMF is to determine reduced relay sets to achieve more efficient multicast transmission in dynamic topologies. To accomplish this, SMF supports the ability to modify its multicast packet forwarding rules based on the relay set state, which is dynamically determined. In this way, SMF forwarding can continue to operate effectively when the topology changes.

B.2.1 Selection Methods

Classical Flooding is the simplest case of SMF multicast forwarding. Each forwarding node is required to rebroadcast a packet when heard for the first time. This approach is extremely simple and only requires DPD and a basic forwarding mechanism. However, it is well known that Classical Flooding results in a significant number of redundant transmissions, which is often referred to as the broadcast storm problem [18]. In MANET environments, reducing unnecessary channel contention significantly improves network performance. Therefore, reducing the number of relay nodes is an important design goal for this environment. Many distributed methods of dynamically electing reduced relay sets that attempt to optimize the flooding of routing control messages in MANET routing peers have been proposed, i.e. Source-specific Multipoint Relay (S-MPR), Non-source-specific MPR (NS-MPR), Essential Connecting Dominating Set (E-CDS), MPR-based CDS (MPR-CDS). Detailed descriptions of these algorithms are as follows:

b) Source-specific Multipoint Relay (S-MPR)

The S-MPR forwarding algorithm is specified for the use in the OLSR protocol. It enables individual nodes to select relays from their set of neighboring nodes using 2-hop topology information. Nodes select a subset of their bi-connected 1-hop neighbors as MPR nodes. This subset provides flooding to all 2-hop neighbors. An S-MPR node forwards if and only if

- It receives a unique multicast packet from any of its bi-connected neighbors.
- The neighbor the packet was received from has selected the node as an MPR.

Thus, S-MPR requires DPD and previous hop knowledge to determine if the packet should be relayed. It is shown in [69] that S-MPR can guarantee minimal hop paths between all nodes in the network, while maintaining an efficient connected dominated set.

c) Non-source-specific MPR (NS- MPR)

NS-MPR enables flooding techniques that do not require previous hop information during the forwarding. It combines all source-specific selected MPRs into a common relay node set. In this case, a node forwards every unique packet if and only if

- It has been selected as an MPR by any other node regardless of the previous hop of the packet.

Thus, NS- MPR removes the requirement of packet previous hop knowledge needed for S-MPR while maintaining minimal hop path forwarding.

d) Essential Connecting Dominating Set (E-CDS)

A dominating set (DS) in a network graph is a set of vertices whose neighbors constitute all the vertices in the graph. A connected DS (CDS) is a DS forming a connected graph. E-CDS allows nodes to use 2-hop neighborhood topology information to dynamically select themselves as relay nodes to form a CDS. Similar to NS-MPR, E-CDS does not require the previous hop knowledge for packet forwarding. Nodes select themselves as relays using neighborhood priority information. Priority values do not need to be unique and can be a combination of values such as power level and address values. Although uniqueness is not required, duplicate values can result in a greater number of forwarders. In order to correctly assign themselves as relays, priority values need to be learned within 2-hop neighbors. E-CDS nodes select themselves as relays if and only if

- The node's priority is greater than all its 2-hop neighbors,
- or there is no path from the highest priority neighbor to all other 1 and 2-hop neighbors using only nodes with greater priorities as relays.

Once a node has selected itself as a relay, all unique multicast packets are rebroadcast. Unlike SMPR, E-CDS does not guarantee minimal hop paths between nodes.

e) MPR-based CDS (MPR-CDS)

Similar to NS-MPR, MPR-CDS also has no requirement for previous hop knowledge. But MPR-CDS has properties similar to both E-CDS and S-MPR. It reduces the number of forwarding nodes to a more efficient subset of MPRs than the NS-MPR approach described above. MPR-CDS requires that nodes know unique network addresses for each node within their 2-hop neighbors. After neighbor discovery, a node using MPR-CDS will forward all unique packets if and only if

- The node's identifier is higher than all its 1-hop neighbors,
- or it has been selected as an MPR by the node that has the highest identifier in its 1-hop neighbors.

B.2.2 Method Comparison

[77] presented the design of a SMF prototype and some initial performance results using the NRL mobile network emulation system and various optional flooding approaches

within the design framework. Comparing the total multicast overhead as a function of time, topology, and representative flooding algorithms, Classical Flooding has the highest overhead in all possible topologies compared to all the other algorithms. The overhead of S-MPR decreased as the number of relay nodes is reduced. Comparing the total multicast goodput, Classical Flooding demonstrated the worst maximum achievable performance regardless of topology. For S-MPR, the maximal attainable throughput decreased as the required relay set increased. Other results also showed that Classical Flooding occasionally had a slight robustness gain at various times at the expense of redundant transmissions. S-MPR has the smallest size of the relay set than ECDS and NS-MPR. Unlike S-MPR, the NS-MPR technique, implemented to eliminate previous hop routing dependencies in the forwarding decision, did not achieve high efficiency under scaled network conditions.

Considering the multicast efficiency of SMF, we choose it as one of the four protocols in the performance comparison in the Section 6. Since S-MPR has relatively higher throughput and less number of forwarders, which makes it a more efficient relay set selection algorithm, we will use SMF based on S-MPR.

APPENDIX C CodeBCast

In CodeBCat, two algorithms that rely only on local 2-hop topology information and make use of opportunistic listening were proposed to reduce the number of transmissions: 1) a simple XOR-based coding algorithm 2) a Reed-Solomon based coding algorithm.

C.1 XOR-based Algorithm

For the XOR-based algorithm, each node *u* with a set of native packets *P* in its output queue finds a subset of native packets *Q* to XOR so that a neighbor *v* of *u* should be able to decode the XOR-ed packet *p* using stored native packets. The XOR-based greedy algorithm takes the packet *p* at the head of the queue and then looks for other packets in the queue that can allow all neighbors of node *u* to decode the packet when combined with *p*. If successful, these packets are added to the coded packet. Once such a coded packet is received by a node, it can extract the missing packet by XOR-ing the received packet with the remaining packets it already received and buffered. This requires that the coded packet carries with it an indication that it is a coded packet, and the unique identifiers of its contributing native packets. In case that the neighbor reception table of a node *u* was inaccurate, a node will be unable to successfully decode a packet. Also a node may not gain any new information from this packet if it already received all constituent native packets.

C.2 Reed-Solomon Code based Algorithm

For the Reed-Solomon based coding algorithm, let *P* be the ordered set of *N* native packets in *u*'s output queue. Let P_v be the set of packets *v* has received, for each $v \in N(u)$. Let $k = \max\{|P - P_v|, v \in N(u)\}$. Let $\theta = \begin{pmatrix} 1^0 & 2^0 & ... & n^0 \\ 1^1 & 2^1 & ... & n^1 \\ ... & ... & ... & ... \\ 1^k & 2^k & ... & n^k \end{pmatrix}$ be the *k*n* Vandermonde matrix where 1, 2, …, *n* are labels of elements in the finite field. The minimal number of encoded packets that needs to be sent so that each neighbor *v* can decode the packets in *(P − P_v)* is *k* and the set of *k* packets are given by $Q = \theta \times P$. Therefore a node constructs the coded packet set $Q = \theta \times P$. It then adds the set of native packet IDs to each coded packet and the index number of codes used. When a node v receives an encoded packet consisting of n native packets (set P), v first goes over all native packets received in its packet pool. It collects P_v, the subset of packets in P that it has already received. It then constructs a decoding matrix and adds the new coefficient vector to this matrix. For each decoded native packet q, node v can now process q.

C.3 Algorithm Comparison

The simulation results [9] showed that the coding-based deterministic approach outperformed the coding-based probabilistic approach and the Reed-Solomon based coding algorithm outperformed the XOR-based algorithm in terms of the coding gain and the packet delivery ratio.

CodeBCast relies only on local 2-hop topology information and makes extensive use of opportunistic listening to reduce the number of transmissions. The set of packets that are grouped together to achieve coding gains is local to each node and thus the generation management and decoding delay can be avoided. Since CodeBCast is an implemented protocol which has higher coding gain and packet delivery ratio than the non-coding protocol and the code of CodeBCast exists, we chose CodeBCast in the performance comparison.