# Title

## Implementation of Energy-Efficient Routing Protocols for Mobile Ad hoc Networks (MANET)

**By:**

**Dereje Girma**

A Thesis submitted to the Faculty of

Technology

Of

Addis Ababa University

in partial fulfillment of the requirements for

the degree of

Master of Science

In

Computer Engineering

Department of Electrical and Computer Engineering

August, 2004

Implementation of Energy-Efficient Routing protocols for MANETs

This Thesis entitled:

Implementation of Energy Efficient Routing Protocols for MANETs

written by Dereje Girma

has been approved for the Department of Electrical and Computer Engineering

| | | |
|---|---|---|
| _____ | _____ | _____ |
| Advisor | Signature | Date |
| _____ | _____ | _____ |
| Chairman, Dept's Graduate Committee | Signature | Date |
| _____ | _____ | _____ |
| Chairman, Faculty's Graduate Committee | Signature | Date |
| _____ | _____ | _____ |
| Dean, Graduate School Graduate Committee | Signature | Date |

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

# Acknowledgements

This Thesis has been a joint effort of my advisors Mr. Yacob Astatke, Dr. Mohammed Abdo and Dr. Raimond Kumudha. My greatest gratitude is reserved for Mr. Yacob Astatke. He will always have the honor of introducing me to the world of research. This thesis is a product of his never-ending knowledge, ingenious ideas, constant encouragement, and abundant patience. He has helped in every stage beginning from the inception of the idea, carrying out the experiments, accomplishing the results, and finally documenting the thesis. I would like to thank him inadequately for everything. His determination, work ethics, and the ability to keep things simple yet effective have been my inspiration.
He has helped me become a better researcher and more important a better person.

On the personal front, I would like to deeply thank my family for all the support, and encouragement. I would like to thank my parents for inspiring me to pursue higher education.

# Table of Contents

# Tables

# Figures

# Abstract

The nodes in a mobile ad hoc network also form its routing infrastructure. Previous research shows that the idle power consumption in the nodes is significant, as the network interfaces on them is always on in order to maintain the routing fabric. As mobile nodes are dependent on battery power, there is a need for protocols that minimize energy consumption. In this thesis, the Dynamic Source Routing protocol used for on-demand routing in an ad hoc network is modified to reduce the power consumption in nodes by adaptively putting their interfaces to sleep. In an ad hoc network, it is impossible to predict accurately when it is all right for a node to put its network interface to sleep, using only its own information. In the approach presented, the time slot during which the interface is on is alternated with a time slot during which the interface is put to sleep. The duration of the on period depends only on indigenously available information about the number of packets the interface receives during this time slot. In the absence of any network activity in the on slots, the sleep period is linearly increased up to a maximum. The report explains all the factors that can affect the performance of the modified routing protocol and its influence on the energy consumption in the network. The penalty of increase in delay and packet loss is unavoidable and the levels of the factors are identified to minimize the penalty. The modified protocol is implemented in the Ns-2 network simulator. A linear equation is used to model the energy consumption for each node in the network. Simulations are conducted to test the modified protocol and the factors varied to study their impact. The results are compared with those obtained from the simulations using the unmodified DSR protocol. The results show average energy savings per-packet of up to 25% with an average of 2-3ms per-packet increase in the delay. The packet loss is comparable to the unmodified DSR protocol.

# Chapter 1

# 1. Introduction

## 1.1 Overview

"Ad hoc" in Latin means "for this". In English, it is an adjective meaning "For the particular purpose in hand or view". An ad hoc network is defined as a network that comes together as needed without the assistance of existing infrastructure. An ad hoc network consists of laptops and/or PDAs that exchange data when brought together. In contrast to infrastructure based wireless networks, in ad hoc networks all nodes are mobile and can be connected dynamically in an arbitrary manner. This may be done either because it may not be economically practical or physically possible to provide the necessary infrastructure or because the situation does not permit its installation. Some classic examples would be situations where friends or business associates would run into each other in an airport terminal and wish to exchange business cards, or in case of an emergency, a group of rescue workers may need to be quickly deployed. In such situations, a collection of mobile hosts with wireless network interfaces may form a temporary network without the aid of any established infrastructure or centralized administration. This type of wireless network is known as an **ad hoc network.**

In the case where only two hosts, within the transmission range, are involved in the ad hoc network, no real routing protocol or routing decisions are necessary. But in many practical ad hoc networks, two hosts that wish to correspond may not close enough to be within wireless transmission range of each other. These hosts could communicate if other hosts between them also participating in the ad hoc network are willing to forward packets for them.

As an example, consider the figure shown below. Mobile host *C* is not within the range of host *A*'s wireless transmitter (indicated by the circle around *A*) and host *A* is not within the range of host *C*'s wireless transmitter. Now, if *A* and *C* wish to interact with each other, they may in this case procure the services of host *B* to forward packets for them, since host *B* lies within the transmission range of both *A* and *C*. A real ad hoc network may be more complicated than this example, due to the inherent non uniform propagation characteristics of wireless transmissions and due to the possibility that any or all of the hosts involved may move at any time.

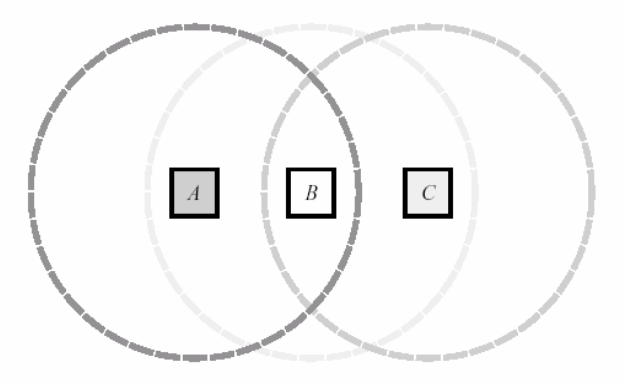



**Figure 1.1 Host A and C are 'connected' via host B.**

From a graph theory, we can view an ad hoc network as a graph, G(A, E(t)), which is formed by denoting each mobile host by a node and drawing an edge between the two nodes if they are with in the wireless communication range of each other. The set of edges (E(t)), is denoted as a function of time, as it keep changing as the node in the ad hoc networks move around. The topology can be very arbitrary since there are no restrictions on where the nodes are located and how they move with respect to each other.

We consider in this thesis the extension to multi-hop ad hoc networks, where the intermediate nodes forward packets from source to destination nodes when the source and destination nodes are unable to communicate directly. Each node in the network acts as both an end-host and as a router due to the limited propagation range of each node's wireless transmission. The nodes in the network are mobile in nature where the nodes could either be identical or heterogenous.

A typical ad hoc network consists of nodes that are usually battery operated devices such as laptops, PDAs or sensor nodes that come together and spontaneously form a network. Energy conservation is a critical issue as the lifetime of these nodes depends on the life of the system. Research has been carried out to conserve energy at various levels i.e., at the hardware level, operating system, application level. Research has proved that it is critical to focus on the inter-dependencies that exist among layers of the protocol stack. In this thesis, we present an implementation of a cross-layered protocol that employs an energy aware metric to route packets and bring about energy conservation. This implementation is a first of its kind; previous work in developing such protocols has been limited to theory.

## 1.2 Historical Development

Packet networking evolved in the 1960s where streams of data moving from one computer to another were reduced to smaller units called "packets" and were given headers containing the required routing information. Computers on the network examine these headers and move the data packets through the network towards the final destination. These networks were self-healing as the routing intelligence was distributed among the network nodes.

Work began in 1969 on an actual packet network, called ARPANET, which developed and implemented a set of protocols called TCP/IP. The ARPANET ultimately evolved into today's Internet. In 1970, radio technology and packet networks were brought together for the first time, and a radio network called ALOHANET, based at the University of Hawaii, was put into operation. The ALOHANET spurred active research in peer-to-peer packet radio networks. This work was targeted at military communications and was expensive.

Another wireless networking technology that was being developed was cellular communications. This concept was conceived by Bell Laboratories in the 1960s and 1970s. AMPS was the first US cellular telephone system that deployed in 1983 by Ameritech in Chicago, IL. In 1993, AMPS offered data services over the cellular network through the cellular digital packet data (CDPD) service. Currently Global System for Mobile Communications (GSM) and Code Division Multiple Access (CDMA) cellular networks offer data services. These wireless data networks are proprietory in nature, do not currently offer high data rates, and depend on service providers to develop the expensive infrastructure.

Work in the Industrial Scientific and Medical (ISM) band led to the evolution of the 802.11 standard which offers wireless data services at speeds of 2 Mbps and greater. Using low cost radio devices currently 802.11b has 15 million users worldwide and the technology is still evolving by adding extensions to the existing protocol. These networks consists of an Access Point (AP) that acts as a centralized controller and clients associate with the AP, which acts as a router to route traffic among the clients and acts as a gateway to the outside world. These networks are hence static by nature and depend on infrastructure for deployment. Next generation applications demand rapid deployment, high bandwidth, and easy reconfiguration.

Current cellular and 802.11b networks are unable to meet these demands. The wide availability of low cost 802.11 and similar radios has led researchers to revisit the early peer-to-peer models. Such reconfigurable, rapidly deployable peer-to-peer networks we denote ad hoc networks.

## 1.3 Types of wireless communications

This section makes precise the types of wireless networks and the role of ad hoc networks.
At the link layer, wireless networks can be classified as:

- Point to Point networks:

  A point-to-point network is the simplest form of wireless network, composed of two radios in direct communication with each other. Examples of such networks are microwave links.

- Point to Multi-Point networks:

  A point-to-multipoint network is a shared link between a central radio and client radios at multiple sites. Examples of such networks are broadcast networks and early mobile phone services.

- Multi-Point to Multi-Point networks:

  These networks mirror the structure the wired Internet. A typical example could be a communication network that consists of walkie-talkies.

At the network layer, wireless networks can be classified as

- Fixed Infrastructure networks:

  These networks employ a centralized architecture where a central node acts as a gateway to wired infrastructure for other wireless nodes in the network. All the traffic is routed through this central node. At the link layer these are point to multi-point. This architecture is typical in case of cellular networks, wireless LANs, and satellite networks.

- Peer to Peer networks:

  These networks are decentralized in nature where each node acts as a router and

communication can take place directly between nodes in the wireless transmission range of each other. At the link layer, these are multi-point to multi-point.

In case the source and the destination nodes are out of transmission range, an intermediate node in the transmission range of both nodes acts as a relay node and routes the packets between the source and destination node. We focus on a special case: so called multi-hop networks. This type of peer to peer network we denote as an ad hoc network. This is restricted to peer-to-peer communication when all nodes are within range of each other and so is more restrictive, thus our definition.

The motivation for deploying ad hoc networks is described in the next section. In 802.11b, there is a so called ad hoc mode.

## 1.4 Applications of ad hoc networks

Ad hoc networks are specifically designed to cater to a particular application. This section discusses potential applications to motivate the reasons for deploying ad hoc networks. The essential characteristic of an ad hoc network is the ability of forming spontaneous networks between nodes that are in range of each other. This is a feature of a number of military, commercial, and social applications

### 1.4.1 Military Application

Military applications require the war fighters and their mobile platforms to be able to move freely without any restrictions imposed by wired communication devices. These applications should thus be self configuring, independent of any centralized control stations, and should be infrastructure independent in nature. These networks need to be robust in nature, i.e., they should not have a single point of failure. Ad hoc networks are thus an appropriate solution for such applications.

### 1.4.2 Commercial Application

The lack of infrastructure in ad hoc networks is a motivating factor for deployment in commercial applications as it reduces the cost of infrastructure investments. Ad hoc networks also have other

commercial advantages due to the ease of network reconfiguration and reduced maintenance costs. Examples of commercial applications are as follows:

- Collaborative Networks

  A typical application of a collaborative ad hoc network can be considered a conference room with participant's wishing to communicate with each other without the mediation of global Internet connectivity. In such a scenario, a collaborative network can be set up among the participants' devices. Such networks involve exchange of data between devices such as laptops, palmtops, PDAs, and other information devices. Each participant can thus communicate with any other participant in the network without requiring any centralized routing infrastructure. These networks are thus collaborative in nature and are useful in cases where business network infrastructure is often missing or in scenarios where reduction in the cost of using infrastructure links is important.

- Home Networks

  These networks involve communications between PCs, laptops, PDAs, cordless phones, smart appliances, and entertainment systems in and around the home. Peer-to-peer communication among these devices will reduce the overhead of going through a centralized node and thus makes ad hoc networks a natural choice for implementing home networking applications.

- Distributed Control Systems

  Ad hoc wireless networks allow distributed control with remote plants, sensors and actuators linked together through wireless communication. These networks help in co-coordinating unmanned mobile units and lead to a reduction in maintenance and reconfiguration costs. Ad hoc wireless networks are used to co-ordinate the control of multiple vehicles in an automated highway system, coordination of unmanned airborne vehicles, and remote control of manufacturing units.

## 1.4.3 Community Networks

The concept of a general purpose ad hoc network is identified as a step toward next-generation ad hoc network development. An open community network is a novel information infrastructure for local communities based on wireless multi-hopping technologies, which may support an advanced information-oriented society in the twenty-first century. A community network consists of one or more computers providing services to people using computers and terminals to gain access to those services and to each other. Community network terminals can be set up at

public places like libraries, bus stations, schools, Laundromats, community and senior centers, social service agencies, public markets, and shopping malls. Community networks can also be accessible from home via computers and, increasingly, from the Internet. Such networks are excellent example.

This section discussed the applications of ad hoc networks that range from military applications, commercial applications, and the newly forming community networks that are considered to be next generation ad hoc networks.

## 1.5 Energy Conservation in ad hoc networks

Many of the devices described in the previous section are battery operated and thus energy constrained. For instance, a typical battery used in laptops and PDAs with wireless adapter has a lifetime of two hours. Nodes in an ad hoc network share a symbiotic relationship where each node acts as an end host as well as a router. Thus, each node carries out its individual processing as well as acts as a forwarding node, thus expending energy in processing and forwarding of packets. This reduces the lifetime of the nodes in an ad hoc network. Energy conservation is thus critical in such networks. This section dwells on the issue of energy conservation in an ad hoc network consisting of such energy constrained devices.

Traditional laptops and Personal Digital Assistants (PDAs) are possible choices for mobile devices used in a MANET. The significant consumers of power in a typical laptop are the CPU, liquid crystal display, I/O subsystem (hard disk, system memory, keyboard/mouse, CD-ROM) and the wireless network interface card. A typical example from a Toshiba 410 CDT mobile computer demonstrates that nearly 36% of power consumed is by the display, 21% by the CPU/memory, 18% by the wireless interface and 18% by the hard drive. Consequently, current hardware design of mobile microprocessors, displays and disks has taken energy conservation into account. PDAs, an example of low power CPUs, typically do not have the large display and the large capacity disk drives of laptops. An example of a PDA is the Compaq's experimental pocket computer that has an idle power consumption of 100 - 200mW. In contrast, the ideal power consumption of a Lucent IEEE 802.11 WaveLAN 2 Mbps interface card is about 1W. As noted above, since a node in a MANET is also a router, it generally spends a significant percentage of time in the idle state when it is neither sending nor receiving data traffic but waiting to forward data. Hence, the need for minimizing power consumption is even stronger in devices such as PDAs. In general, incorporating energy conservation strategies in the design of

other layers of the protocol stack and in the hardware design of the wireless network interface card is extremely important.

Research has been carried out to conserve energy at various levels in a system; at the application level, applications have been designed that adapt on the basis of the current energy level of the system. At the operating system level, energy conservation can be brought about by switching systems to an idle or stand-by mode. Energy is conserved at the CPU level by reducing the clock speed and voltage level of the CPU. At the MAC level, schemes that power down the cards when not in use bring about energy savings. Energy can be conserved at the routing level according to by designing cross layered protocols and deploying low power routing algorithms that use power cost of the route as a metric for routing packets.

## 1.6 Research Overview

In this thesis, the Dynamic Source Routing (DSR) protocol, an on-demand routing protocol used in a MANET, is modified to reduce the total energy consumption in the network. We propose a simple algorithm in which the interface on a node in a MANET saves energy by periodically putting its network interface in the low power sleep state in the absence of network activity– sending, receiving or forwarding data packets. In the sleep state, a network interface on a node cannot send, receive or forward data packets. A linear equation is presented to model the energy consumption by a network interface on a node. We spell out the factors and their levels affecting the performance of the modified protocol and their impact on the energy consumption. The adverse impacts of the modified protocol are analyzed. In order to minimize the negative effects, the optimal levels of the factors are identified through the process of experimentation. This report presents the results of the detailed simulations comparing the relative performance of the modified protocol with that of unmodified DSR. The results establish the effectiveness of the algorithm presented in this report in reducing energy consumption in a network. The linear equation used in the report to model the energy consumption by an interface is also used to calculate the maximum energy saving possible in a simulation. The calculated maximum value is compared with the result from the simulation and the difference between the two is analyzed for a subset of the simulations.

Section 2 presents a brief overview of the types of routing protocols used in a MANET and gives a detailed description of DSR–the main component under study in this report. Section 3 presents about a different mobility models which describes the movement of the mobile nodes. Section 4 presents the proposed algorithm and lists the services provided by a MANET running the

modified routing protocol and also discusses about the Simulation setup. The possible outcomes of making the modifications are discussed. Subsequently, the metrics used in comparing the performance of the protocol with that of DSR are selected. Section 5 discusses about the structure and functionalities of the Network Simulator (NS). Section 6 discusses about the Simulation results. The report ends with conclusions in Section 6.

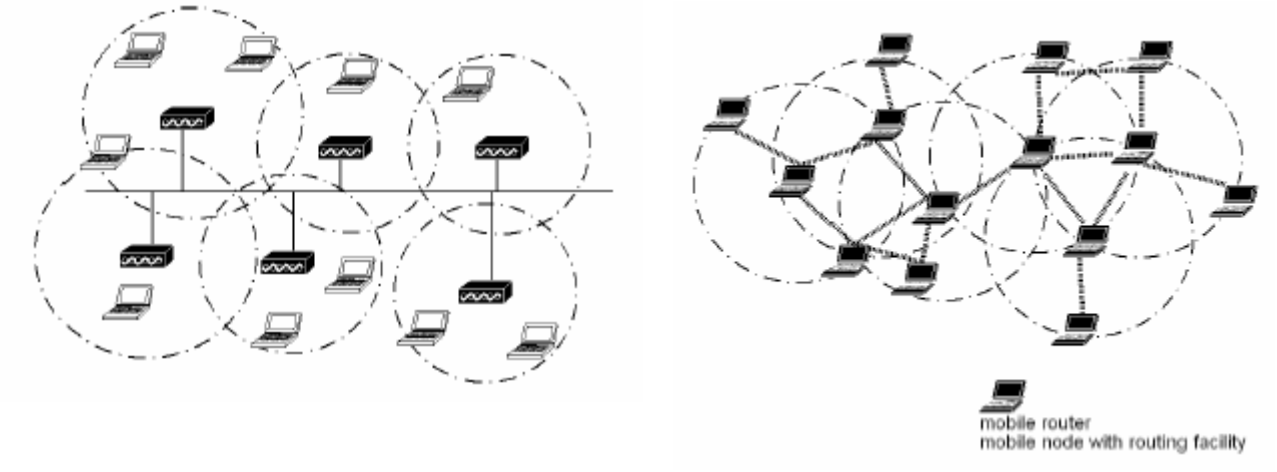
# Chapter 2

# 2. Literature Review

In this chapter we discuss the different ad hoc routing protocols, provide a comparison between them and finally enumerate research work that has been carried out in the design, implementation, and testing of energy aware ad hoc routing protocols.

## 2.1 Mobile Ad hoc Networks

A mobile ad-hoc network (MANET) is a collection of nodes, which have the possibility to connect on a wireless medium and form an arbitrary and dynamic network with wireless links. That means that links between the nodes can change during time, new nodes can join the network, and other nodes can leave it. A MANET is expected to be of larger size than the radio range of the wireless antennas, because of this fact it could be necessary to route the traffic through a multi-hop path to give two nodes the ability to communicate. Additionally, wireless links have significantly lower capacity and transmission range than their hardwired counterparts due to effects of signal fading, noise, and interference conditions. Consequently, multiple hops may be needed for one node to exchange data with another across the network. Thus, each node operates as both a host as well as a router, forwarding packets for other mobile nodes. Most importantly, some or all the nodes in a MANET may rely on battery power. There are neither fixed routers nor fixed locations for the routers as in cellular networks - also known as infrastructure networks (Fig. 1(a)). Cellular networks consist of a wired backbone which connects the base-stations. The mobile nodes can only communicate over a one-hop wireless link to the base-station, multi-hop wireless links are not possible. By contrast, a MANET has no permanent infrastructure at all. All mobile nodes act as mobile routers. A MANET is depicted in Fig. 1(b).




**(a) Infrastructure Network** **(b) Mobile Ad-hoc Network**

**Figure 2.1 Wireless Network Structures**

## 2.2 Protocol Stack

In this section the protocol stack for mobile ad hoc networks is described. This gives a comprehensive picture of, and helps to better understand, mobile ad hoc networks. Fig 2.1 shows the protocol stack which consists of five layers: physical layer, data link layer, network layer, transport layer and application layer. It has similarities to the TCP/IP protocol suite. As can be seen the OSI layers for session, presentation and application are merged into one section, the application layer.

On the left of Figure 2.1, the OSI model is shown. It is a layered framework for the design of network systems that allows for communication across all types of computer systems.

In the middle of the figure, the TCP/IP suite is illustrated. Because it was designed before the OSI model, the layers in the TCP/IP suite do not correspond exactly to the OSI layers. The lower four layers are the same but the fifth layer in the TCP/IP suite (the application) layer is equivalent to the combined session, presentation and application layers of the OSI model.

On the right, the MANET protocol stack – which is similar to the TCP/IP suite - is shown. The main difference between these two protocols stacks lies in the network layer. Mobile nodes (which are both hosts and routers) use an ad hoc routing protocol to route packets. In the

physical and data link layer, mobile nodes run protocols that have been designed for wireless channels. In the simulation tool used in this project, the standard IEEE 802.11 is used in these layers.

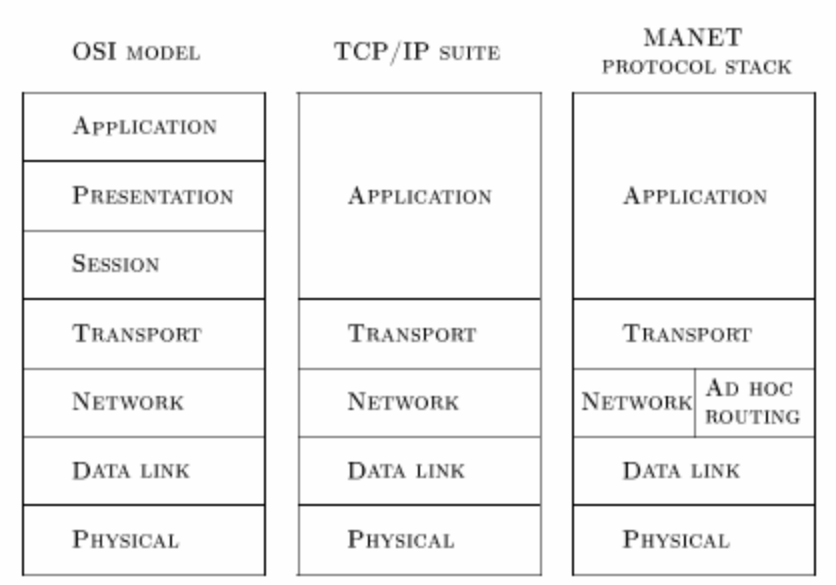



**Figure 2.2 The OSI Model, TCP/IP suite and MANET protocol stack**

This thesis focuses on ad hoc routing which is handled by the network layer. The network layer is divided into two parts: Network and Ad Hoc Routing. The protocol used in the network part is Internet Protocol (IP) and the protocol used in the ad hoc routing part is Dynamic Source Routing (DSR). Other Ad hoc routing protocols that can be used in this part of the network layer are discussed in Sections 2.4, 2.5 and 2.6. One of the reasons to why DSR has been used in this study will be described in the next Section.

In the transport layer the User Datagram Protocol (UDP), is used in this study. The Transmission Control Protocol (TCP) is not used because there are some researches showing that TCP does not perform well in mobile ad hoc networks. One reason to this is that in wired networks, lost packets are almost always due to congestion but in mobile ad hoc networks lost packets are more often caused by other reasons like route changes or transmission errors.

## 2.3 Why do we need specialized Ad hoc routing protocols

An ad hoc network is a co-operative network of mobile nodes that communicate over a wireless medium. Ad hoc networks differ significantly from existing networks. First, the topology of the

nodes in the network is dynamic. Second, these networks are self-configuring in nature and require de-centralized control and administration. Such networks do not assume all the nodes to be in the direct transmission range of each other. Hence these networks require specialized routing protocols that provide self-starting behavior. Energy constrained nodes, low channel bandwidth, node mobility, high channel error rates, and channel variability are some of the limitations in an ad hoc network. Under these conditions, existing wired network routing protocols would fail or perform poorly. Thus, ad hoc networks demand specialized routing protocols. In this section, we will discuss the different types of ad hoc routing protocols.

Ad hoc routing protocols are classified based on the manner in which route tables are constructed, maintained, and updated. They are classified as

- Table-driven
- Source initiated or demand-driven

Developed for the same underlying ad hoc network, the characteristic of these protocols differ significantly. The following sections describe the protocols and group them according to their characteristics.

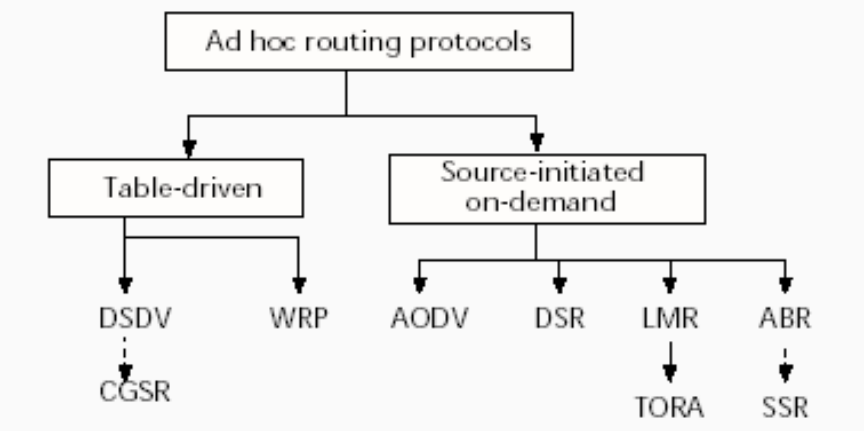



**Figure 2.3 Categorization of Ad-Hoc Routing Protocol.**

## 2.4 Proactive Routing Protocols

Table-driven routing protocols maintain consistent, up-to-date routing information from each node to every other node in the network. These nodes maintain routing tables and respond to the changes in the network topology by propagating updates throughout the network in order to maintain a consistent view of the network. The different table-driven protocols differ in the

number of routing tables and the methods by which changes in the network structure are broadcast. Keeping routes to all destinations up-to-date, even if they are not used, is a disadvantage with regard to the usage of bandwidth and of network resources. Periodic broadcasts of routing updates attempt to keep the routing table completely updated. The rate of these broadcast messages reflects the rate at which the topology of the network changes even if no traffic is affected by the change. This implies that the use of scarce resources – power and link bandwidth will increase with increased node mobility. In general, proactive routing protocols are unsuitable for deployment in a MANET. Proactive protocols do not scale in the frequency of topology change. Therefore the proactive strategy is appropriate for a low mobility network.

The following sections discuss the different table-driven ad hoc routing protocols.

### 2.4.1 Destination Sequenced Distance Vector Protocol (DSDV)

The Destination Sequenced Distance Vector Protocol (DSDV) is a proactive, distance vector protocol which uses the Bellmann-Ford algorithm. Compared to RIP one more attribute is added to the routing table. The sequence number as new attribute guarantees loop-freedom. It makes it possible for the mobile to distinguish stale routes from new ones and that is how it prevents loops. DSDV can only handle bidirectional links.

#### Routing Table Management

The routing table in each node consists of a list of all available nodes, their metric, the next hop to destination and a sequence number generated by the destination node. The routing table is used to transmit packets through the ad hoc network. In order to keep the routing table consistent with the dynamically changing topology of an ad hoc network the nodes have to update the routing table periodically or when there is a significant change in the network. Therefore mobile nodes advertise their routing information by broadcasting a routing table update packet. The metric of an update packet starts with metric one for one-hop neighbors and is incremented by each forwarding node and additionally the original node tags the update packet with a sequence number. The receiving nodes update their routing tables if the sequence number of the update is greater than the current one or it is equal and the metric is smaller than the current metric. Delaying the advertisement of routes until best routes have been found may minimize fluctuations of the routing table. On the other hand the spreading of the routing information has to be frequent and quick enough to guarantee the consistency of the routing tables in a dynamic

network. There exist two types of update packets. One is the full dump which contains the entire routing table and must be periodically exchanged. The other is an incremental update which only consists of the information changed since the last full dump.

### Responding to Topology Changes

DSDV responds to broken links by invalidating all routes that contain this link. The routes are immediately assigned an infinite metric and an incremented sequence number. Broken links can be detected by link and physical layer components or if a node receives no broadcast packets from its next neighbors for a while. Then the detecting node broadcasts immediately an update packet and informs the other nodes with it. If the link to a node is up again, the routes will be re-established when the node broadcasts its routing table.

## 2.4.2 Optimized Linkstate Routing Protocol

The Optimized Linkstate Protocol is a proactive link state routing protocol. It uses periodic messages for updating the topology information. OLSR is based on the following mechanisms:

- Neighbor sensing based on periodic exchange of HELLO messages
- Efficient flooding of control traffic using the concept of multipoint relays
- Computation of an optimal route using the shortest-path algorithm

### Neighbor Sensing

Neighbor sensing is the detection of changes in the neighborhood of the node. Node **A** is called neighbor of node **B** if the two nodes are directly linked, allowing data transmission in both directions of the link. The node **C** is called a two-hop neighbor of **A**, if node **C** is not neighbor of node and there exists a symmetric link between **A** and **B** and a symmetric link between **B** and **C**. For neighbor sensing the node periodically emits HELLO messages. The HELLO message consists of the emitting node's address, the list of his neighbors, including the link status (e.g. asymmetric or symmetric). A node thereby informs its neighbors of which neighbors it has confirmed communication.

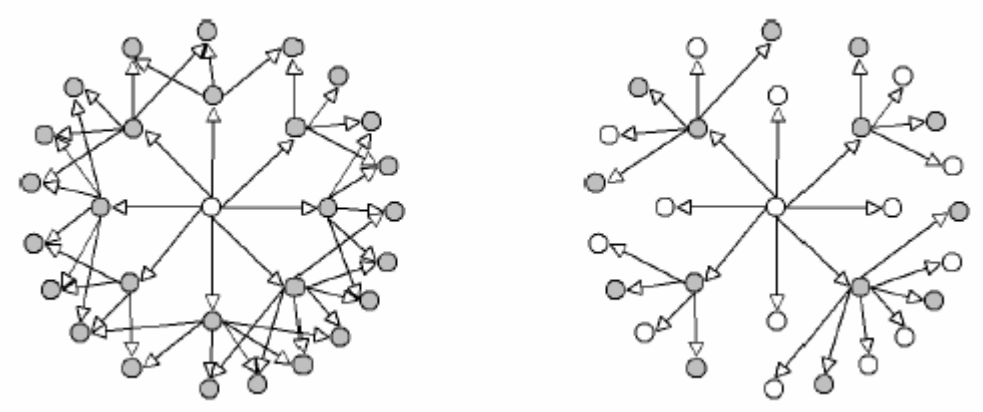
By receiving a HELLO message, a node generates information describing its two-hop neighborhood and the quality of the links in its neighborhood. Each node maintains this

information set which is valid for a limited time only and has to be refreshed to keep it valid.

## Message Flooding and Multipoint Relays

HELLO-messages are exchanged between neighbors only. These messages provide topology information for the nodes. Because the size of a MANET can be considerable, there is a need for efficient distribution of topological information in a network of any size. The task is to provide a mechanism which allows spreading information to each node without unnecessary, duplicate retransmissions.

The multipoint relay (MPR) concept decreases the flooding overhead in contrast with full flooding.



**(a) Pure flooding** **(b) Multipoint Relays**

**Figure 2.4 Comparison of two flooding techniques**

**Full flooding** A node retransmits broadcast packet after reception of its first copy, further duplicate receptions are dropped and not forwarded (Fig. 2(a)).

**MPR flooding** Each node chooses independently a set of nodes as MPRs (multipoint relays). For this purpose it utilizes the information about its two-hop neighbors to get a minimal MPR set. This set is chosen so that a node reaches all its two-hop neighbors through its MPR relays. Each node maintains a list of nodes which selected it as MPR (MPR selector set). A MPR node only retransmits a broadcast packet if it is received from a node for which it is located in the MPR selector set, further receptions of the same packet are dropped (Fig. 2(b)).

## 2.5 Reactive Routing Protocols

This type of routing creates routes only when desired by the source node. The source node initiates a process called **route discovery** when it requires a route to the destination. This process is completed when a route is found or when all the possible routes are examined. The process of **route maintenance** is carried out to maintain the established routes until either the destination becomes unavailable or when the route is no longer required. As reactive routing protocols flood the network to discover the route, they are not optimal in terms of bandwidth utilization, but they scale well in the frequency of topology change. Thus this strategy is suitable for high mobility networks.

### 2.5.1 Dynamic Source Routing

Dynamic Source Routing, DSR, is a reactive routing protocol that uses source routing to send packets. It is reactive like AODV which means that it only requests a route when it needs one and does not require that the nodes maintain routes to destinations that are not communicating. It uses source routing which means that the source must know the complete hop sequence to the destination.

Each node maintains a route cache, where all routes it knows are stored. The route discovery process is initiated only if the desired route cannot be found in the route cache. To limit the number of route requests propagated, a node processes the route request message only if it has not already received the message and its address is not present in the route record of the message.

As mentioned before, DSR uses source routing, i.e. the source determines the complete sequence of hops that each packet should traverse. This requires that the sequence of hops is included in each packet's header. A negative consequence of this is the routing overhead every packet has to carry. However, one big advantage is that intermediate nodes can learn routes from the source routes in the packets they receive. Since finding a route is generally a costly operation in terms of time, bandwidth and energy, this is a strong argument for using source routing. Another advantage of source routing is that it avoids the need for up-to-date routing information in the intermediate nodes through which the packets are forwarded since all necessary routing information is included in the packets. Finally it avoids routing loops easily because the complete route is determined by a single node instead of making the decision hop-by-hop.

### Route Discovery

Route Discovery is used whenever a source node desires a route to a destination node. First, the source node looks up its route cache to determine if it already contains a route to the destination. If the source finds a valid route to the destination, it uses this route to send its data packets. If the node does not have a valid route to the destination, it initiates the route discovery process by broadcasting a route request message. The route request message contains the address of the source and the destination, and a unique identification number.

An intermediate node that receives a route request message searches its route cache for a route to the destination. If no route is found, it appends its address to the route record of the message and forwards the message to its neighbors. The message propagates through the network until it reaches either the destination or an intermediate node with a route to the destination. Then a route reply message, containing the proper hop sequence for reaching the destination, is generated and unicasts back to the source node.

### Route Maintenance

Route Maintenance is used to handle route breaks. When a node encounters a fatal transmission problem at its data link layer, it removes the route from its route cache and generates a route error message. The route error message is sent to each node that has send a packet routed over the broken link. When a node receives a route error message, it removes the hop in error from its route cache.

Acknowledgment messages are used to verify the correct operation of the route links. In wireless networks acknowledgments are often provided as e.g. an existing standard part of the MAC protocol in use, such as the link-layer acknowledgment frame defined by IEEE 802.11. If a built-in acknowledgment mechanism is not available, the node transmitting the message can explicitly request a DSR-specific software acknowledgment to be returned by the next node along the route.

## 2.5.2 Ad hoc On-Demand Distance Vector (AODV)

Ad hoc On-Demand Distance Vector, AODV, is a distance vector routing protocol that is reactive. The reactive property of the routing protocol implies that it only requests a route when it needs

one and does not require that the mobile nodes maintain routes to destinations that are not communicating. AODV guarantees loop-free routes by using sequence numbers that indicates how new, or fresh, a route is.

AODV require each node to maintain a routing table containing one route entry for each destination that the node is communicating with. Each route entry keeps track of certain fields. Some of these fields are:

- **Destination IP address**: The IP address of the destination for which a route is supplied
- **Destination Sequence Number**: The destination sequence number associated with the route.
- **Next Hop**: Either the destination itself or an intermediate node designated to forward packets to the destination.
- **Hop Count**: The number of hops from the Originator IP Address to the Destination IP Address.
- **Lifetime**: The time in milliseconds for which nodes receiving the RREP consider the route to be valid.
- **Routing Flags**: The state of the route; up (valid), down (not valid), in repair.

## Route Discovery

Whenever a source node desires a route to a destination node for which it does not already have a route, it broadcasts a route request (RREQ) message to all its neighbors. The neighbors update their information for the source and create a reverse route entries for the source node in their routing tables. A neighbour receiving a RREQ may send a route reply (RREP) if it is either the destination or if it has an unexpired route to the destination. If any of these two cases is satisfied, the neighbour unicasts a RREP back to the source. Along the path back to the destination, intermediate nodes that receive the RREP create forward route entries for the destination node in their routing tables. If none of the two cases mentioned is satisfied, the neighbour rebroadcasts (forwards) the RREQ.

Each mobile node keeps a cache where it stores the source IP address and ID of the received RREQs during the last PATH_DISCOVERY_TIME seconds (see Appendix). If a mobile node receives another RREQ with the same source IP address and RREQ ID

during this period, it is discarded. Hence, duplicated RREQs are prevented and not forwarded.

When searching for a route to the destination node, the source node uses the expanding ring search technique to prevent unnecessary network-wide dissemination of RREQs. This is done by controlling the value of the time to live (TTL) field in the IP header. The first RREQ message sent by the source has TTL=TTL_START (see Appendix). This value of TTL defines the maximal number of hops a RREQ can move through the mobile ad hoc network, i.e. it decides how far the RREQ is broadcasted. In other words, it implies that the RREQ which is broadcasted by the source is received only by mobile nodes TTL hops away from the source (and of course all mobile nodes less than TTL hop away from the source). Besides setting the TTL, the time out for receiving a RREP is also set. If the RREQ times out without reception of a corresponding RREP, the source broadcasts the RREQ again. This time TTL is incremented by TTL_INCREMENT, i.e. the TTL of the second RREQ message is TTL_START + TTL_INCREMENT. This continues until a RREP is received or until TTL reaches TTL_THRESHOLD. If TTL reaches TTL_THRESHOLD a RREQ is sent with TTL=NET_DIAMETER, which disseminate the RREQ widely, throughout the MANET. A RREQ that is broadcasted with TTL=NET_DIAMETER is referred to as a *network-wide search*. If a source node does a network-wide search and still does not receive a RREP, it may try again to find a route to the destination node, up to a maximum of RREQ_RETRIES times.

## Route Maintenance

When a link in a route breaks, the node upstream of the break invalidates all its routes that use the broken link. Then, the node broadcasts a *route error* (RERR) message to its neighbors (TTL is set to one). The RERR message contains the IP address of each destination which has become unreachable due to the link break. Upon reception of a RERR message, a node searches its routing table to see if it has any route to the unreachable destination(s) (listed in the RERR message) which uses the source of the RERR as the next hop. If such routes exist, they are invalidated and the node broadcasts a new RERR message to its neighbors. This process continues until the source receives a RERR message. The source invalidates the listed routes as previously described and reinitiates the route discovery process if needed.

## 2.6 Proactive and Reactive Routing Protocols

Table-driven protocols have the overhead of route updates with no regard to the frequency of forwarding packets that take place in the ad hoc network. The routing information is constantly propagated within the network. This is not the case with on-demand protocols where routing information is exchanged only when the source wishes to send some information to the destination and has no information about the destination in its route cache. On the other hand, since routing information is constantly propagated and updated in table-driven protocols, information about a particular source destination route is always available regardless of whether or not this information is required. This feature leads to significant signaling overhead and power consumption.

Since both battery and bandwidth are scarce resources in ad hoc networks, this becomes a serious limitation. From the discussion of table-based protocols provided in Section 2.4 and on demand protocols presented in Section 2.5, we conclude that table-based protocols incur significantly high routing overhead, etc. and hence lead to increase the energy consumption compared to the on-demand protocols. In the next section, we will discuss the research work that has been carried out to reduce energy consumption in ad hoc networks.

## 2.7 Prior work involving reduction in energy consumed in ad hoc networks

The protocol stack designed for a MANET has to deal with the complex problems inherent to the wireless channel due to mobility of nodes. Additionally, the design must also consider energy efficiency. The consideration for energy efficiency has been given attention in all the layers of the protocol stack. The survey of energy efficient network protocols for wireless networks in [2] provides an extensive summary of recent work in all the layers of the protocol stack.

[3] gives the experimental measurements of per-packet energy consumption of a Lucent IEEE 802.11 WaveLAN PC 2 Mbps card operating in a MANET. The study measures energy cost associated with broadcast and unicast traffic at the sender, the receiver and the non-destination nodes. The per-packet energy consumption model is reported in this study as a collection of linear equations at the MAC layer. The linear energy consumption model for a node, instead of a per-packet model proposed in [3], presented in this thesis is based on the above model. The

measured values of the WaveLAN card (listed in Table 1) are used to calculate the energy cost to a node in the simulations done for this study.

| | Measured Current | Power |
|---|---|---|
| Sleep mode | 14 mA | 0.0664 W |
| Idle mode | 178 mA | 0.844 W |
| Receive mode | 204 mA | 0.967 W |
| Transmit mode | 280 mA | 1.327 W |
| Power supply | 4.74 V | |

**Table 2.1 Lucent IEEE 802.11 WaveLAN PC 2 Mbps Card Characteristics**

The focus of this study was DSR–a network layer protocol. [4] presents a simulation based comparison of DSR and Ad hoc On Demand Vector (AODV) routing protocols in a MANET. [4] uses the per-packet energy consumption model reported in [3]. However, instead of using actual measurements as was done in [3], extensive post-processing of the Ns-2 generated log files was used to calculate the cost for sending, receiving and discarding packets. [4] found that DSR is more efficient than AODV in terms of packets sent and received. The authors also used extensive post-processing to find the per-packet energy cost in DSR once the promiscuous mode of the network interfaces on nodes using DSR is disabled. The authors labeled this variant of DSR as DSR-np. [4] also found that DSR-np is the most energy efficient protocol. DSR and DSR-np are explained in more detail latter in this section. Unlike the model used in [3,4], this thesis also considers the cost associated with a transition by a node to a low power sleep state.

The simulation environment and the methodology used in [4] were first presented in [6]. The simulations (including post-processing to calculate the energy costs) done for this study are conducted using a similar environment and methodology as given in [4,6]. In all of the above publications except the survey in [2], the basis of routing is to route packets through a path with a minimum number of hops. There is no consideration given for routing packets through a path that requires minimum energy. However, in [7], the authors focus on designing a protocol to reduce energy consumption to increase the lifetime of each node and consequently increase network life as well. Consequently, shortest hop routing is no longer applicable. Instead, other criteria such as the remaining lifetime of the battery on a node are considered to make energy efficient routing decisions. The above approach however requires the dissemination of the location of every node and the remaining battery power throughout the network. This requirement creates an increase in communication and computation overhead and the use of

non-shortest hops could potentially cause an increase in delay at the receiver. The authors in [7] though found no extra delay in using the power-aware metric while the energy cost per-packet was significantly reduced. However, the nodes did not move randomly and the authors opined that in the presence of random movement, the cost savings would be small. In addition, the protocol only addresses routing of unicast traffic with respect to battery power consumption. The simulations done in this study consider random motion, deal with both unicast and broadcast traffic in DSR, and nodes require no additional information to make decisions about saving energy.

[8] also states that the existing DSR implementations are not energy efficient. DSR does not have any mechanism to exchange the packet transmit power information in order to use transmit power control. In order to implement minimum energy routing, the authors in [8] argue for modifications to DSR, IEEE 802.11 MAC layer protocol and for changes in the power circuits in the radio hardware to do power control over the transmission medium. In contrast, no such modifications are required by the approach presented in this thesis.

Another approach to building an energy-efficient routing protocol in a MANET is the Span protocol presented in [9]. Span uses a distributed, randomized algorithm where nodes make local decisions on whether to sleep or to join the routing backbone as a coordinator. The decision by a node to join as a coordinator is based on an estimate on the number of neighbors benefiting from it being awake as well as energy available to it. The role of a coordinator is similar to that of a base station in an infrastructure wireless network. It buffers and schedules traffic to the nodes in its region, allowing those nodes to spend as much time as possible in a low-power consumption sleep state. In Span, geographic forwarding was chosen to route packets. The sender and receiver nodes in [9] do not move making the use of geographic forwarding simple. Like the requirement of dissemination of battery power in [7], in this approach too, the nodes need to know the location of other nodes in the region. Besides, [3] states that rapid connectivity changes in a clustering protocol like Span may require expensive recomputations of cluster memberships. The computation cost of the protocol processing and the cost of a geographic positioning system in Span has the potential to become a significant source of energy consumption. Table 2 lists the power consumption of a commercially available GPS receiver. However, working with the simplification that the sender and receiver nodes do not move and ignoring the computation cost and the energy consumption cost of a positioning system, the authors in [9] found that the approach was successful in reducing the energy consumption by 50%. The energy savings presented in Section 8 of this report are only half as those presented in [9]. However, the work presented in this thesis does not impose any restriction on the mobility of the sender and receiver

nodes. In addition, the algorithm presented in this study does not incur the overhead of a geographic positioning system and the computation costs associated with a cluster based approach like Span.

| | Measured Current | Power |
|---|---|---|
| Continuous mode | 140 mA | 0.462 W |
| Power save mode (at 1 update per second) | 50 mA | 0.165 W |
| Operating Voltage | 3.3 V | |

**Table 2.2 Power consumption of a GPS-MS1E Miniature GPS Receiver**

In general, the basic idea is to derive an energy efficient strategy that allows a network interface card on a node in a MANET to remain in the low-power sleep state as long as possible. This thesis presents an algorithm to achieve energy savings using this basic idea.

Research has been carried out at each layer of the OSI stack to minimize energy consumption.
At the physical layer, research is carried out addressing the energy problem considering two views)

- An increase in battery capacity, and
- A decrease in the amount of energy consumed at the wireless terminal.

There has not been a significant breakthrough recently in battery-technology recently and hence this goal can only be achieved by decreasing the energy consumed in the wireless terminal. Low-power design at the hardware layer can be achieved by incorporating techniques like variable clock speed CPUs.

At the link layer, Researcher suggests using a scheme that uses a small packet size for registration and bandwidth request and thus reduces the energy consumption. By using small packets, in the EC-MAC protocol, the collisions that occur during registration and reservation are reduced. They propose a scheme by which the schedule for data transmission of each mobile is broadcast. This enables the mobiles to switch to standby mode until the receive start time. Another solution that is proposed is to turn off the transceiver whenever the node determines that it will not be receiving data for a period of time. The PAMAS protocol uses this approach. For MANETs, the routing layer has to carry out the functionality of routing under the conditions of mobility. Researchers focus on designing protocols to reduce energy consumption to increase the life of each mobile, thus increasing network life as well. This is carried out by defining metrics such as Energy Consumed per packet, Time to network partition, Variance in power levels across

mobiles, Cost per packet, and Maximum mobile cost. This results in a shortest-cost routing protocol with respect to the metrics.

[13] describe a series of experiments carried out to model the energy consumption of wireless Ethernet cards operating in an ad hoc environment. The author classifies the different energy consumption costs based on broadcast traffic, point-to-point traffic, discard traffic, promiscuous mode operation, and idle mode operation. They present an empirical energy model of these costs which is derived from measurements using the Lucent IEEE 802.11b Wave LAN card and discuss the implications of the energy model in designing energy aware ad hoc routing protocols. Our work refers to the energy model proposed in this paper.

At the routing layer, [14] suggests a minimum power routing scheme that has been designed using the table-driven approach. [15] brings about power savings by setting the transmit power of all nodes to control the topology of a multi-hop wireless network. [16] provides a distributed position based network protocol that minimizes energy consumption in mobile wireless networks. The power consumption model consists of large scale and small scale channel variations, path loss, transmit power, receive power, and the power required to process the signal. This implementation is well-suited for networks that consist of stationary nodes. The authors do not present results about signaling overhead due to mobility. The energy model assumes negligible costs for packet reception and processing.

[17] propose algorithms to select routes and corresponding power levels in a static wireless ad hoc network such that the system lifetime (in terms of battery life) is maximized. [18] suggest a location information based energy conserving routing algorithm that works above the ad hoc routing agent protocol to bring about energy conservation in the network by powering down intermediate nodes while still maintaining connectivity. [19] develop an online approximate power aware routing algorithm to maximize the lifetime of the ad hoc network, which involves choosing between the minimal power consumption path and the path that maximizes the minimum residual power in the network. [20] suggest a distributed coordination technique for a multi-hop ad hoc wireless network that reduces energy consumption without significantly reducing the capacity and connectivity of the network. Energy savings are achieved by keeping certain chosen coordinator nodes active while the other nodes in the network are in a power save mode. [21] proposes a table-based scheme that strives to maximize the end-to-end throughput by fixing transmit power levels for nodes in a cluster.

[22] present a critique between the two on-demand ad hoc routing protocols by stressing the differences in their dynamic behaviors that lead to the factors that affect their performance. They also discuss the interpretation of the simulation results carried out to compare the performance of these protocols based on the packet delivery fraction, average end-to-end delay of data packets, normalized routing load, and normalized MAC load. The simulation model was based on ns-2 and used the DCF of IEEE 802.11b as the MAC protocol. The simulation results indicated that DSR outperforms AODV in less stressful (lower load and or mobility and smaller number of nodes) situation when delay and throughput were considered. However in case of a more stressful situation, AODV outperforms DSR. This could be attributed to the aggressive route caching that is carried out by DSR. DSR uses unicast routing packets which were expensive for the 802.11 MAC layer that was used and hence DSR's apparent savings on routing load did not translate to the expected reduction on the real load on the network.

[23] present a protocol for power control in ad hoc networks by providing a solution to satisfy three objectives of maximizing the traffic carrying capacity of the entire network, extending battery life by using low power routes, and reducing contention at the MAC layer. The protocol called, COMPOW, aims to operate all nodes at a common power level which is selected to be the smallest power level at which the network remains connected. The solution also provides sub-optimal traffic carrying capacity of the network, produces power aware routes, and reduces MAC layer contention. This protocol can be used as a plug and play feature with proactive routing protocols. It maintains multiple routing tables at the user level which incurs an additional bookkeeping overhead. The paper does not address mobility issues in mobile ad hoc networks and does not provide performance results.

[23] propose a general solution for enhancing operating systems with new services to support ad hoc routing, and an On Demand Routing Module (ODRM). The ODRM is implemented as a module and contains a library which is called the Ad hoc Support Library (ASL). Packets with known destination are routed using the kernel routing tables. The route discovery API in the ASL is initiated when packets with unknown destination arrive. The TUN/TAP is used to copy packets to user space to maintain the ad hoc routing table that exists in user space. On finding the destination, the ASL has API's that update the kernel routing table; these update packets from user space are re-injected into the kernel using raw sockets. It describes an AODV implementation using this infrastructure and also discusses other AODV implementations. However, such an infrastructure is not suitable for source routing protocols as the kernel routing tables do not provide mechanisms to store source routes.

At the operating systems level, energy consumption can be carried out by efficient scheduling to manage access to physical resources like CPU, memory and disk space from the applications running on the host. Researchers at Intel Corporation propose to reduce power dissipation in CPUs by designing portable devices that can be operated at lower speeds by scaling down the supply voltage. Another technique mentioned in the same paper is predictive shutdown during periods of inactivity.

At the application layer, development of APIs such as Advanced Configuration and Power Interface and Power Management Analysis tools can assist in creating programs that are power-conserving.

## 2.8 Modeling Energy Consumption in MANET

In this study, the link layer of the network protocol stack uses the IEEE 802.11 Medium Access Control (MAC) protocol. The transmission of each unicast data packet is preceded by an exchange of Request-to-Send/Clear-to-Send (RTS/CTS) control packets that reserve the wireless channel for transmission of the data packet. Each correctly received unicast packet is followed by the transmission of an acknowledgment (ACK) control packet to the sender. The sender retransmits the packet a small number of times until the ACK is received. The number of retry attempts by the MAC layer in Ns-2 is 4. After the small number of unsuccessful retries, the packet is dropped. Broadcast packets are sent only when carrier sense indicates that the medium is clear, but they are not preceded by an RTS/CTS exchange and are not acknowledged by the recipients

### 2.8.1. Energy Consumption states

In a MANET, wireless communication involves usage of a transceiver at the source, intermediate, and destination nodes. The transmitter sends control, route request and response, as well as data packets originating at or routed through the transmitting node. The receiver is used to receive data and control packets - some of which are destined for the receiving node and some of which are forwarded. A wireless network interface has five possible energy consumption states (six including the off state). Transmit and receive states are for transmitting and receiving data, control and routing packets. In the idle state, which is the default state for ad hoc environment, the interface can transmit or receive packets. The sleep state has extremely low power consumption as the interface can neither transmit nor receive in this state. Lastly, a card can enter a reduced energy discard state while the media carries uninteresting traffic. The decision to enter

the reduced energy discard state is made by the non-destination nodes in the range of the sender. This decision is based on the packet size information in the RTS control packet that is exchanged between the sender and the receiver at the start of packet transfer. The nodes in the range of the receiver, but not the sender (hidden terminals) do not use the above strategy to save energy. The reduced energy state uses slightly less power than the idle state, but significantly more than that used in the sleep state. As explained in the previous section, for this study, the reduced energy discard state was not considered. Moreover, the network interfaces were operated in promiscuous mode to enable the use of all optimizations in the DSR protocol.

## 2.8.2 Analytical Model for Energy Consumption in a Node

The total energy consumed by the network interface on a node can thus be modeled by a linear equation:

**Total energy** (in Joules) = $(P_T \times T_T) + (P_R \times T_R) + (P_S \times T_S) + (P_I \times T_I)$ **(1)**

where $P_T$, $P_R$, $P_S$, $P_{I\ are}$ the values for power consumed in the transmit, receive, sleep and idle states respectively by a node in Watts.

Table 2.1 lists these values for a Lucent IEEE 802.11WaveLAN PC 2 Mbps card. These values are used for the quantitative analysis of the results of the simulations done in Section 8. $T_T$, $T_R$, $T_S$, $T_I$ are the summation of times in seconds during which a node is in transmit, receive, sleep and idle states respectively during an experiment run. [3] also presents a linear equation describing the energy consumed by the network interface when a host sends, receives or discards a packet. It is:

**Energy** = m x size + b **(2)**

where m is the incremental component proportional to the size of the packet and b is the fixed component associated with device state changes and channel acquisition.

Experimental results presented in [3] confirm the correctness of the linear model.

Equation (1) is similar to equation (2), as both are linear. Equation (1) deals with the incremental cost component (m) since the time spent by a node transmitting or sending is proportional to the size of data. The time spent in transmit or receive state also includes part of the fixed cost dealing

with channel acquisition. However, it does not deal with the part of the fixed cost component (b) dealing with time spent in changing states. According to [2], the turnaround time between transmit and receive state is only between 6 to 30 µs and is ignored in this study. [11] notes that in a WaveLAN card, the turnaround time between sleep and idle state is more than 250 µs. During the turnaround period, the card already has the power consumption of the idle state. In this study, the times spent by a node in transmit, receive and sleep states were recorded.

The time spent in the sleep state is recorded after subtracting the above mentioned turnaround time. For the remaining time of the experiment run, the node was considered to be in the idle state. For DSR, $T_S$ is zero. In the modified DSR presented in the next section, $T^m_S$ is not zero. For a fair comparison, the total time in an experiment run is same for both the cases. Thus, the total time is given by:

**Total time T** (in seconds) = $T_T + T_R + T_I = T^m_T + T^m_R + T^m_I + T^m_S$ **(3)**

where $T_T$, $T_R$, $T_I$ are the summation of times in seconds during which a node is in transmit, receive and idle states respectively during an experiment run with a MANET deploying DSR. $T^m_T$, $T^m_R$, $T^m_I$ are the corresponding time periods for a node in a MANET deploying the modified DSR and $T^m_S$ is the total time during which a node was in the sleep state.

The fraction of energy saved from equation (1) is given below:

$$\frac{P_T * (T_T - T^m{}_T) + P_R * (T_R - T^m{}_R) + P_I * (T_I - T^m{}_I) - P_s * T^M{}_S}{(P_T * T_T) + (P_R * T_R) + (P_I * T_I)} \quad \textbf{(4)}$$

From (3), $T^m_S = T_T - T^m_T + T_R - T^m_R + T_I - T^m_I$ **(5)**

In an ideal scenario, a node would alternate between the sleep state and the idle state and always correctly be in transmit or receive state, i.e. $T_T - T^m_T$ and $T_R - T^m_R$ is zero.

Equation (5) is simplified to: $T^m_S = T_I - T^m_I$ **(6)**

Similarly, equation (4) is simplified to: $\frac{(P_I - P_S) * T^M{}_S}{(P_T * T_T) + (P_R * T_R) + (P_I * T_I)}$ **(7)**

Multiplying (3) by $(P_I - P_S)$ gives: $(P_I - P_S) \times (T_T + T_R + T_I) = (P_I - P_S) \times T$ **(8)**

Because $P_I - P_S \approx P_I$ and $P_T > P_I$ and $P_R > P_I$, it can be concluded that:

$$(P_T \times T_T) + (P_R \times T_R) + (P_I \times T_I) > (P_I - P_S) \times (T_T + T_R + T_I) \quad \textbf{(9)}$$

Applying equation (8) and (9) to (7) gives:

$$\frac{(P_I - P_S) * T^M{}_S}{(P_T * T_T) + (P_R * T_R) + (P_I * T_I)} < \frac{T_S{}^M}{T} \quad \textbf{(10)}$$

Equation (10) gives the maximum value of fraction of energy saved in a node. It will be used to validate the results obtained from post-processing the log files generated by the simulations in Ns-2 in Section 5.

# Chapter 3

# 3. Energy Aware Implementation Issues

In this chapter, we discuss the mobility model, energy model used, factors that can be exploited to reduce the energy consumed by the existing ad-hoc routing protocols, choice of routing protocol, and the required features for an energy aware routing protocol.

## 3.1 Mobility Models

To evaluate the performance of a protocol for an ad hoc network, it is necessary to test the protocol under realistic conditions, especially including the movement of the mobile nodes. A survey of different mobility models follows. This includes the Random Waypoint Model that is used in this research.

### 3.1.1 Random Walk Mobility Model

This model is based on random directions and speeds. By randomly choosing a direction between 0 and 2π and a speed between 0 and Vmax, the mobile node moves from its current position. A recalculation of speed and direction occurs after a given time or a given distance walked. The random walk mobility model is memoryless. Future directions and speeds are independent of the past speeds and directions. This can cause unrealistic movement such as sharp turns or sudden stops. If the specified time or distance is short, the nodes are only walking on a very restricted area on the simulation area.

### 3.1.2 Random Waypoint Mobility Model

A mobile node begins the simulation by waiting a specified pause-time. After this time it selects a random destination in the area and a random speed distributed uniformly between *0* m/s and Vmax. After reaching its destination point, the mobile node waits again pause-time seconds before choosing a new way point and speed. The mobile nodes are initially distributed over the simulation area. This distribution is not representative to the final distribution caused by node

movements. To ensure a random initial configuration for each simulation, it is necessary to discard a certain simulation time and to start registering simulation results after that time.

The Random Waypoint Mobility Model is very widely used in simulation studies of MANET. The performance measures in mobile ad-hoc networks are affected by the mobility model used. One of the most important parameters in mobile ad-hoc simulations is the nodal speed. The users want to adjust the average speed to be stabilized around a certain value and not to change over time. They also want to be able to compare the performance of the mobile ad-hoc routing protocols under different nodal speeds. For the Random Waypoint Mobility Model a common expectation is that the average is about half of the maximum, because the speeds in a Random Waypoint Model are chosen uniformly between 0 m/s and Vmax. In the Random Waypoint Mobility Model a node selects its destination and its speed. The node keeps moving until it reaches its destination at that speed. If it selects a far destination and a low speed around 0 m/s, it travels for a long time with low speed. If it selects a speed near Vmax the time traveling with this high speed will be short. After a certain time the node has traveled much more time at low speed than at high speed. The average speed will approach 0 m/s. The suggestion to prevent this problem is choosing, e.g. 1 m/s instead of 0 m/s as Vmin. With this approach the average speed stabilizes after a certain time at a value below 1/2 * Vmax.

### 3.1.3 Random Direction Mobility Model

To reduce density waves in the average number of neighbors by the Random Waypoint Model the Random Direction Mobility Model was created. *Density waves* are the clustering of nodes in one part of the simulation area. For the Random Waypoint Mobility Model the probability of choosing a location near the center or a way point which requires traveling through the center of the area is high. The Random Direction Mobility Model was invented to prevent this behavior and to promote a semi-constant number of neighbors. The mobile node selects a direction and travels to the border of the simulation area. If the boundary is reached, the node pauses for a specific time and then chooses a new direction and the process goes on. Because of pausing on the border of the area, the hop count for this mobility model is much higher than for most other mobility models.

## 3.2 Energy Efficient Routing

Routing is one of the key issues in MANETs due to their highly dynamic and distributed nature. In particular, energy efficient routing may be the most important design criteria for MANETs since mobile nodes will be powered by batteries with limited capacity. Power failure of a mobile node not only affect the node itself but also its ability to forward packets on behalf of others and thus the overall network lifetime. For this reason, many research efforts have been devoted to developing energy aware routing protocols.

Based on the aforementioned discussions, this paper surveys and classifies numerous energy efficient routing mechanisms proposed for MANETs. They can be broadly categorized based on *when* the energy optimization is performed. A mobile node consumes its battery energy not only when it actively sends or receives packets but also when it stays idle listening to the wireless medium for any possible communication requests from other nodes. Thus, energy efficient routing protocols minimize either the *active* communication energy required to transmit and receive data packets or the energy during *inactive* periods.

For protocols that belong to the former category, the active communication energy can be reduced by adjusting each node's radio power just enough to reach the receiving node but not more than that. This *transmission power control approach* can be extended to determine the optimal routing path that minimizes the total transmission energy required to deliver data packets to the destination. For protocols that belong to the latter category, each node can save the inactivity energy by switching its mode of operation into *sleep/power-down mode* or simply turns it off when there is no data to transmit or receive. This leads to considerable energy savings, especially when the network environment is characterized with low duty cycle of communication activities. However, it requires well-designed routing protocol to guarantee data delivery even if most of the nodes sleep and do not forward packets for other nodes. Another important approach to optimizing active communication energy is *load distribution approach*. While the primary focus of the above two approaches is to minimize energy consumption of individual nodes, the main goal of the load distribution method is to balance the energy usage among the nodes and to maximize the network lifetime by avoiding over-utilized nodes when selecting a routing path.

While it is not clear that any particular algorithm or a class of algorithms is the best for all scenarios, each protocol has definite advantages/disadvantages and is well-suited for certain situations. However, it is possible to combine and integrate the existing solutions to offer a more energy efficient routing mechanism. Since energy efficiency is also a critical issue in other network layers, considerable efforts have been devoted to developing energy-aware MAC and transport protocols. Each layer is supposed to operate in isolation in layered network architecture

but, as some recent studies suggested, the *cross-layer design* is essential to maximize the energy performance. In fact, many routing protocols introduced in this paper use the same concept, i.e. they exploit lower layer mechanisms such as transmission power control and sleep mode operation in their routing layer algorithms.

In contrast to simply establishing correct and efficient routes between pair of nodes, one important goal of a routing protocol is to keep the network functioning as long as possible. As discussed in the first few paragraph of this section, this goal can be accomplished by minimizing mobile nodes' energy not only during active communication but also when they are inactive. *Transmission power control* and *load distribution* are two approaches to minimize the active communication energy, and *sleep/power-down mode* is used to minimize energy during inactivity. Table 3.1 shows taxonomy of the energy efficient routing protocols.

Before presenting protocols that belong to each of the three approaches in the following subsections (3.2.1, 3.2.2 and 3.2.3), *energy-related metrics* that have been used to determine energy efficient routing path instead of the shortest one are discussed. They are

- energy consumed/packet,
- time to network partition,
- variance in node power levels,
- cost/packet, and
- maximum node cost.

The first metric is useful to provide the *min-power path* through which the overall energy consumption for delivering a packet is minimized. Here, each wireless link is annotated with the link cost in terms of transmission energy over the link and the min-power path is the one that minimizes the sum of the link costs along the path. However, a routing algorithm using this metric may result in unbalanced energy spending among mobile nodes. When some particular mobile nodes are unfairly burdened to support many packet-relaying functions, they consume more battery energy and stop running earlier than other nodes disrupting the overall functionality of the ad hoc network. Thus, maximizing the network lifetime (the second metric shown above) is a more fundamental goal of an energy efficient routing algorithm: Given alternative routing paths, select the one that will result in the longest network operation time.

| Approach | | Protocols | Goal |
|---|---|---|---|
| Minimize Active Communication Energy | Transmission Power Control (Section 3.2.1) | • Flow Argumentation Routing (FAR)<br>• Online Max-Min (OMM)<br>• Power Aware Localized Routing (PLR)<br>• Minimum Energy Routing (MER) | Minimize the total transmission Energy but avoid low energy nodes. |
| | | • Retransmission-energy Aware routing (RAR)<br>• Smallest Common Power (COMPOW) | Minimize the total transmission energy while considering retransmission overhead or bi-directionality requirements. |
| | Load Distribution (Section 3.2.2) | • Localized Energy Aware Routing (LEAR)<br>• Conditional Max-Min Battery Capacity Routing (CMMBCR) | Distribute Load to Energy rich nodes. |
| Minimize Inactivity Energy | Sleep/Power-Down Mode (Section 3.2.3) | • SPAN<br>• Geographic Adaptive Fidelity (GAF)<br>• Prototype Embedded Networks (PEN) | Minimize Energy consumption during inactivity. |

**Table 3.1 Taxonomy of energy efficient routing protocols.**

However, since future network lifetime is practically difficult to estimate, the next three metrics have been proposed to achieve the goal indirectly. Variance of residual battery energies of mobile nodes is a simple indication of energy balance and can be used to extend network lifetime. Cost-per-packet metric is similar to the energy-per-packet metric but it includes each node's residual battery life in addition to the transmission energy. The corresponding energy-aware routing protocol prefers the wireless link requiring low transmission energy, but at the same time avoids the node with low residual energy whose node cost is considered high. With the last metric, each path candidate is annotated with the maximum node cost among the intermediate nodes (equivalently, the minimal residual battery life), and the path with the minimum path cost, *min-max path*, is selected. This is also referred to as *max-min path* in some protocols because they use nodes' residual battery life rather than their node cost.

## 3.2.1 Transmission Power Control Approach

A routing algorithm essentially involves finding an optimal route on a given network graph here a vertex represents a mobile node and an edge represents a wireless link between two end nodes that are within each other's radio transmission range. When a node's radio transmission power is controllable, its direct communication range as well as the number of its immediate neighbors are also adjustable. While stronger transmission power increases the transmission range and reduces the hop count to the destination, weaker transmission power makes the topology sparse which may result in network partitioning and high end-to-end delay due to a larger hop count.

In order to illustrate the potential benefits of controlling or adjusting transmission power, consider an example shown in Figure 1, which compares two transmission power models: *constant power model* and *variable power model*. If the transmission power is not controllable and thus constant (*pc*) as shown in Figure 1(a), the routing path *S* -> *D* is the shortest and at the same time the most energy efficient path. On the other hand, if the transmission power is controllable, it may be more energy efficient to transmit packets using intermediate nodes because the required transmission power, *p*, to communicate between two nodes has super-linear dependence on distance, *d*, i.e., $p(d) \propto d^2$. For example, in Figure 1(b), the routing path *S* ->*A*->*D* is more energy efficient than the route *S* -> *D* since $p(|SD|) > p(|SA|) + p(|AD|)$. Node *S* conserves energy by lowering its radio power just enough to reach node *A*, but not enough to reach node *D*.

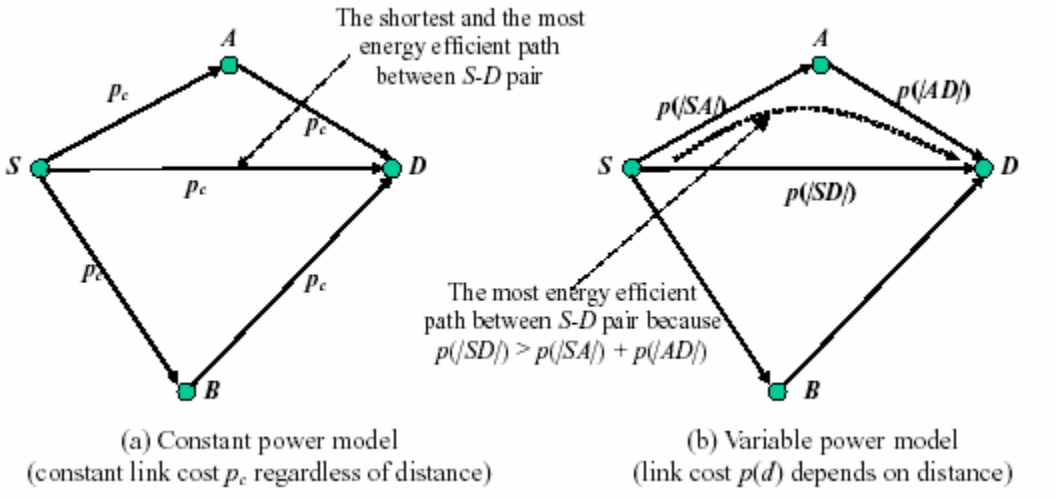



**Figure 3.1 Constant and Variable transmission power model**

There has been active research on topology control of a MANET via transmission power adjustment, and the primary objective is to maintain a connected topology using the minimal

power. Energy efficient routing protocols based on transmission power control find the best route that minimizes the total transmission power between a source-destination pair. It is equivalent to a graph optimization problem, where each link is weighted with the link cost corresponding to the required transmission power (*e.g.*, $p(|SA|)$ for the link $S$ -> $A$). Finding the most energy efficient (*min-power*) route from $S$ to $D$ is equivalent to finding the least cost path in the weighted graph. Section 3.2.1.1 introduces four such routing protocols and Section 3.2.1.2 discusses two link layer issues, such as retransmission overhead and bi-directionality requirement, for implementing the transmission power control approach.

## 3.2.1.1 Transmission Power Optimization

Flow Augmentation Routing (FAR), Online Max-Min Routing (OMM), and Power aware Localized Routing (PLR) protocols fall into this category. Since each node runs the routing algorithm, equivalently the graph optimization algorithm, in a distributed way, it must be supplied with information such as the transmission energy over the wireless link (link cost) and the residual battery energy of the node (reciprocal of node cost). The latter is used to balance the energy consumption by avoiding low energy nodes when selecting a route. The main goal of Minimum Energy Routing (MER) protocol is not to provide energy efficient paths but to make the given path energy efficient by adjusting the transmission power just enough to reach to the next hop node. Table 2 shows the types of information required and the approach used to optimize energy efficiency and avoid low energy nodes.

| Routing Protocol | Required information at each node in addition to that obtained during operation | Approach to optimize energy efficiency and to avoid low energy nodes |
|---|---|---|
| FAR | Link costs of all links<br>Node costs of all nodes<br>Data generation rate at all nodes | • Use graph optimization algorithm<br>• Include node cost in the link cost |
| OMM | Link costs of all links<br>Node costs of all nodes | • Use graph optimization algorithm<br>• Select the max-min path among the number of best min-power paths |
| PLR | Link costs of some links (from itself to its neighbors and to the destination)<br>Node costs of some nodes (all its neighbors) | • Use graph optimization algorithm<br>• Include node cost in the link cost |
| MER | None (Each source node will obtain the link costs through the routing algorithm employed) | • Adjust the transmission power just enough to reach the next hop node in |

| | | |
|---|---|---|
| | | the given routing path |

**Table 3.2 Routing protocols based on transmission power control.**

**FAR (Flow Augmentation Routing) Protocol**

The FAR protocol assumes a static network and finds the optimal routing path for a given source-destination pair that minimizes the sum of link costs along the path. Here, the link cost for link (*i*, *j*) is expressed as $e_{ij}^{X1}E_i^{X2}R_i^{-X3}$, where $e_{ij}$ is the energy cost for a unit flow transmission over the link and *Ei* and *Ri* are the *initial* and *residual energy* at the transmitting node *i*, respectively, and $x1$, $x2$, and $x3$ are nonnegative weighting factors. A link requiring less transmission energy is preferred ($e_{ij}^{X1}$ ). At the same time, a transmitting node with high residual energy ($R_i^{-X}$ ) that leads to better energy balance is also preferred. Depending on the parameters $x1$, $x2$, and $x3$, the corresponding routing algorithm achieves a different goal. For example, with $x1= 0$, $x2 = 0$, and $x3 = 0$, the link cost is always 1 and the optimal path in this case is equivalent to the minimum hop path.

While *eij* and *Ei* are constant for a wireless link (*i*, *j*), $R_i$ continues to drop as communication traffic moves on. An optimal solution at one moment may not be optimal at a later time because $R_i$'s and the corresponding links costs have changed. For this reason, FAR solves the overall optimal solution in an iterative fashion: Solve the optimal route for the first time step, update nodes' residual energy and link costs, and solve another for the next time step, etc. Data generation rate at all nodes during each time step is assumed to be available beforehand.

**OMM (Online Max-Min Routing) Protocol**

FAR maximizes the network lifetime when data generation rate is known. The OMM protocol achieves the same goal without knowing the data generation rate in advance. It optimizes two different metrics of the nodes in the network: Minimizing power consumption (min-power) and maximizing the minimal residual power (max-min). The second metric is helpful in preventing the occurrence of overloaded nodes.

Given all link costs, the OMM protocol first finds the optimal path for a given source-destination pair by using the Dijkstra's algorithm (single-source shortest-path algorithm). This min-power path consumes the minimal power (Pmin) but it is not necessarily the max-min path. In order to optimize the second metric, the OMM protocol obtains multiple near-optimal min-power paths that do not deviate much from the optimal value (i.e., less than zPmin, where $z \geq 1$) and selects the best path that optimizes the max-min metric.

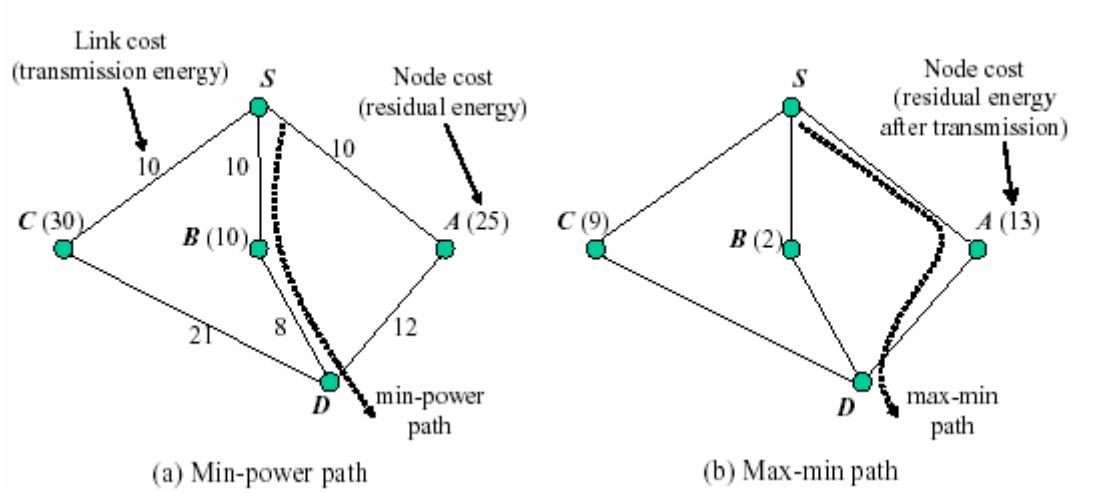



**Figure 3.2 Min-power path and max-min path in the OMM protocol.**

Figure 2 shows an example of the algorithm for a given source (*S*) and a destination (*D*) pair. In Figure 2(a), *S* -> *B* -> *D* is the min-power path as it consumes the minimal energy (*Pmin =18*). If *z=2*, alternative paths *S->A ->D* (path cost=22) and *S->C->D* (path cost=31) can also be considered since their path costs are within the tolerance range (*zPmin =36*). In order to obtain the max-min path among those three path candidates, the node with the minimal residual power in each path must be compared. In this example, each path contains only one intermediate node and thus their residual energies (nodes *A*, *B,* and *C*) are compared. Node *C* has the residual energy of 30 but it will drop to 9 if that path is used to transfer the packets from *S* to *D*. Similarly, nodes *A* and *B* will have the residual energy of 13 and 2, respectively, a shown in Figure 2(b). Therefore, the max-min path among the three min-power paths is *S ->A -> D*.

The parameter *z* measures the tradeoff between the max-min path and the min-power path. When *z*=1, there will not be any alternative path candidate other than the optimal min-power path. Total energy consumption is optimized but energy balance is not considered. When $z=\infty$, all possible paths are considered and the min-power metric is ignored. Therefore, the proper selection of the parameter *z* is important in determining the overall energy performance. A perturbation method is used to adaptively compute *z*. First, an initial value of *z* is randomly chosen, and the residual energy of the most overloaded node, called a *lifetime*, is estimated based on the measurement during a fixed time period of MANET operation. Then, *z* is increased by a small constant, and the lifetime is estimated again after the next time period. If the newly estimated lifetime is longer than the older one, the parameter *z* is increased accordingly; otherwise, *z* is decreased. Since the two successive estimates are calculated based on

measurements during two different time periods, the whole process is based on the assumption that the network traffic distributions are similar as time elapses.

**PLR (Power-aware Localized Routing) Protocol**

Routing algorithms based on global information, such as data generation rate or power level information of all nodes (node costs), may not be practical because each node is provided with only the local information. The PLR protocol is a localized, fully distributed energy aware routing algorithm but it assumes that a source node has the location information of its neighbors and the destination. It is equivalent to knowing the link costs from itself to its neighbors and to the destination. Based on this information, the source cannot find the optimal path but selects the next hop through which the overall transmission power to the destination is minimized.

As discussed previously, a direct communication may consume more energy than an indirect communication via intermediate nodes due to the super-linear relationship between transmission energy and distance. In Figure 3, when node *A* has data packets to send to node *D*, it can either send them directly to *D* or via one of its neighbors (*N*1, *N*2, or *N*3). Note that *A* to *Ni* is a direct transmission while *Ni* to *D* is an indirect transmission with some number of intermediate nodes between *Ni* and *D*. In order to select the optimal route, node *A* evaluates and compares the power consumption of each path candidate. Power consumption of the direct transmission, $p(d)$, can be calculated if the distance is known, i.e., $p(d)=ad^{\alpha}+c$, where $a$ and $c$ are constants, $d$ is the distance between two nodes and $\alpha \geq 2$. It has been shown that power consumption of indirect transmission is minimized when $(n-1)$ equally spaced intermediate nodes relay transmissions along the two end nodes, and the resultant minimum power consumption is $q(d)^2$. Therefore, the node (*A*), whether it is a source or an intermediate node, selects one of its neighbors (*N*1, *N*2, or *N*3) as the next hop node which minimizes $p(|ANi|) + q(|NiD|)$.

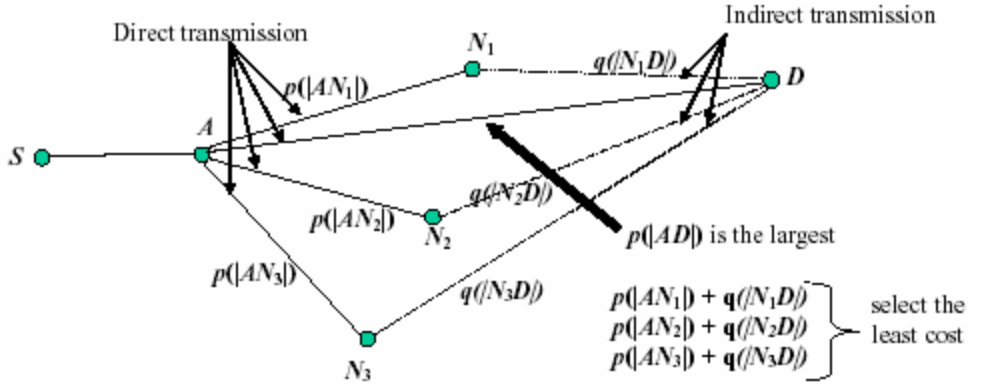



**Figure 3.3 Selection of the next hop node in the PLR protocol.**

**MER (Minimum Energy Routing) Protocol**

The transmission power control approach requires power information such as link costs and node costs. In practice, the following issues need to be addressed: (1) How to obtain accurate power information, (2) how much overhead is associated with the energy aware routing, and (3) how to maintain the minimum energy routes in the presence of mobility.

*Minimum Energy Routing* (MER) protocol addresses these issues and implements the transmission power control mechanism in DSR and IEEE 802.11 MAC protocol with eight selectable options as shown in Table 3. Option A modifies the header of a route-request packet to include the power used by the sender to transmit the packet. The receiving node uses this information as well as radio power level used to receive the packet to calculate the minimum power required for the successful transmission from the sender to itself. This per hop power information is appended at each intermediate node toward the destination and the destination node informs the source node via the route-reply packet. Then, the source node simply inserts this per hop power information in the data packet header so that all the intermediate nodes as well as the source itself transmit the data packet at the controlled power level. Option F applies the same power control mechanism on the MAC layer's ACK packets.

| Options | Implementation level |
|---|---|
| A: Routing packet-based power control | Routing Software / 802.11 firmware |
| B: Minimum Cache Routing | Routing Software |
| C: Cache replies off | Routing Software |
| D: Internal Cache time out | Routing Software |
| E: Multi-hop route discovery | Routing Software |
| F: MAC Layer ACK power control | 802.11 Firmware |
| G: Route maintenance using power sensing of data packets | Routing Software |
| H: MAC level DATA/ACK snooping/gratuitous replies | 802.11 Firmware |

**Table 3.3 Eight options in MER protocol**

Options B, C and D are related to route-cache maintained in the DSR routing algorithm. In Option B, if the source has multiple route candidates in its cache, it calculates the total transmission energy for each possible route based on the power level information obtained via applying Option A and chooses the minimum energy route. In Option G, low energy routes are dynamically adjusted when the required transmission power changes due to node mobility.

Options E and H allow non-participating nodes to snoop on packet exchange and to suggest the sender a more energy efficient route at the routing and the MAC layer, respectively.

### 3.2.1.2 Power Optimization with Other Practical Requirements

As discussed in the previous subsection, the transmission power control is an effective approach to reduce energy consumption in a MANET. However, when applying the technique in routing protocols, some link layer issues need to be considered. This subsection will address these issues.

**Link Error and Retransmission Overhead**

Transmission power control provides an opportunity to save energy by utilizing intermediate nodes between two distant nodes. However, the resultant path with many short-range links may perform worse than a path with fewer long-range links in terms of latency as well as energy consumption. This is because the path with many short-range links would cause more link errors that result in more retransmissions.

Consider a path from a source node $S$ to a destination node $D$ that consists of $N$-1 intermediate nodes indexed as 2, 3, ..., $N$ (the index of the source is 1 and that of the destination is $N$+1). The transmission energy over each link is $p_{i,i+1} = ad_{i,i+1}^{a}$, where $d_{i,i+1}$ refers to the distance between nodes $i$ and $i$+1, $a$ is a constant determined based on the physical environment, and $\alpha \geq 2$. Assuming that each of $N$ links ($L1,2$, $L2,3$, ..., $LN,D$) has an independent link error rate of $e_{i,i+1}$, the number of transmissions (including retransmissions) between node $i$ and node $i$+1 is a geometrically distributed random variable $X$, such that

$$Prob\{X=x\} = e_{i,i+1} \times (1- e_{i,i+1}), \text{ for all} x$$

The mean number of transmissions for the successful transfer of a single packet is thus $1/(1- e_{i,i+1})$. Therefore, the effective transmission energy between nodes $i$ and $i$+1, which includes the effect of the transmission link error, is

$$P_{i,i+1} = p_{i,i+1} \times \frac{1}{1-e_{i,i+1}} = \frac{ad_{i,i+1}^{\alpha}}{1-e_{i,i+1}}$$

When the packet error rate ($e_{i,i+1}$) is not negligible, the benefit of indirect transmission via intermediate nodes can be overshadowed by the inflation factor, $1/(1- e_{i,i+1})$. *Retransmission-Energy*

*Aware Routing* (RAR) protocol modifies the optimization problem with the newly defined link cost to minimize the transmission energy while taking into account the effect of transmission link errors.

**Bidirectionality Requirement**

To deliver packets with minimum energy, the transmission power control approach adjusts each node's radio power and allows different transmission power levels at different nodes. However, in order for the link-level connectivity of a MANET to work correctly, any pair of communicating nodes must share a bidirectional link. For example, at the link level, control packet handshaking is usually employed to enhance the link-level reliability in error-prone wireless environment; i.e., when a node receives a packet, it immediately replies back to the sender with the ACK. If no ACK is returned to the sender, it automatically retransmits the packet. In addition, *RTS (request to send)* and *CTS (clear to send)* packets are exchanged to deal with the *hidden terminal problem*. Therefore, when two nodes have different power levels, data communication along one direction (from the node with stronger transmission power to the other node with weaker transmission power) is possible but not in the reverse direction.

*Smallest Common Power* (COMPOW) protocol presents one simple solution to maintain bi-directionality between any pair of communicating nodes in a MANET. This is achieved by having all the nodes in the MANET maintain a common transmission power level (*Pi*). If *Pi* is too low, a node can reach only a fraction of the nodes in the MANET as in Figure 4(a). If *Pi* is very high, a node can directly reach all other nodes as in Figure 4(b) but results in high energy consumption. In fact, a node can directly or indirectly reach the entire MANET with a smaller *Pi* as shown in Figure 4(c). Therefore, the optimum power level (*Pi*) is the smallest power level at which the entire network is connected.

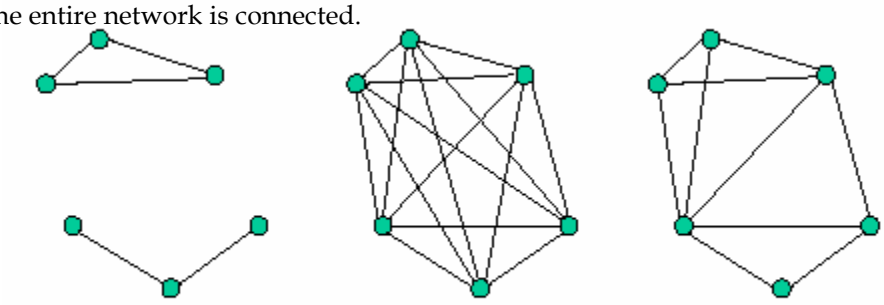



**Figure 3.4 Proper selection of the common transmission power level in COMPOW.**

In COMPOW, it is assumed that the transmission power levels cannot be arbitrarily adjusted but instead it must be selected among a small number of discrete power levels (*P*1, *P*2, ..., *Pmax*). Different power levels result in different node connectivity since they cover different radio transmission ranges. Each node maintains a routing table as in table-driven routing mechanism (see Section 2), but one for each power level ($RT_{P1}$, $RT_{P2}$, …, $RT_{Pmax}$). The number of entries in $RT_{Pi}$, denoted as $|RT_{Pi}|$, means the number of reachable nodes at *Pi*. This includes directly connected nodes as well as indirectly connected nodes via intermediate nodes. By exchanging these routing tables, nodes find the minimal *Pi* that satisfies $|RT_{Pi}|=n$ for all nodes, where *n* is the total number of nodes in the MANET. Extended solutions are also discussed in for the case where there are many discrete power levels and where the latency involved with switching power levels is not negligible.

## 3.2.2 Load Distribution Approach

The specific goal of the load distribution approach is to balance the energy usage of all mobile nodes by selecting a route with underutilized nodes rather than the shortest route. This may result in longer routes but packets are routed only through energy-rich intermediate nodes. Protocols based on this approach do not necessarily provide the lowest energy route, but prevent certain nodes from being overloaded, and thus, ensures longer network lifetime. This subsection discusses two such protocols:
*Localized Energy-Aware Routing (LEAR)* and *Conditional Max-Min Battery Capacity Routing (CMMBR)* protocols.

**Localized Energy Aware Routing (LEAR) Protocol**

The LEAR routing protocol is based on DSR but modifies the route discovery procedure for balanced energy consumption. In DSR, when a node receives a route-request message, it appends its identity in the message's header and forwards it toward the destination. Thus, an intermediate node always relay messages if the corresponding route is selected. However, in LEAR, a node determines whether to forward the route-request message or not depending on its *residual battery power* ($E_r$). When $E_r$ is higher than a *threshold value* ($T_{hr}$), the node forwards the route-request message; otherwise, it drops the message and refuses to participate in relaying packets. Therefore, the destination node will receive a route-request message only when all intermediate nodes along a route have good battery levels, and nodes with low battery levels can conserve their battery power.

LEAR is a distributed algorithm where each node makes its routing decision based only on local information, such as $E_r$ and $T_{hr}$. As $E_r$ decreases as time passes, the value of $T_{hr}$ must also be decreased adaptively in order to identify energy-rich and energy-hungry nodes in a relative sense. For example, if the source node does not receive any reply for a route-request message, the source re-sends the same route-request message. If an intermediate node receives the duplicate request message, it adjusts (i.e., lowers) its $T_{hr}$ to allow forwarding to continue. A sequence number is used to distinguish between the original and the re-sent route-request message.

A complication can arise when route-cache replies are directly sent to the source without evaluating the residual battery levels of all following intermediate nodes. To prevent this from occurring, a new control message, route-cache, is used as shown in Figure 5. In the original DSR, when an intermediate node (node *B*) finds a route in its route cache, it stops broadcast forwarding and sends a route-reply back to the source. However, in LEAR, the intermediate node (node *B*) stops broadcast forwarding the route-request message but continues to forward the route-cache message (*B -> C1 -> C2 -> D* in this example). This does not add any significant traffic to the network because the route-cache message can be delivered in unicast mode.

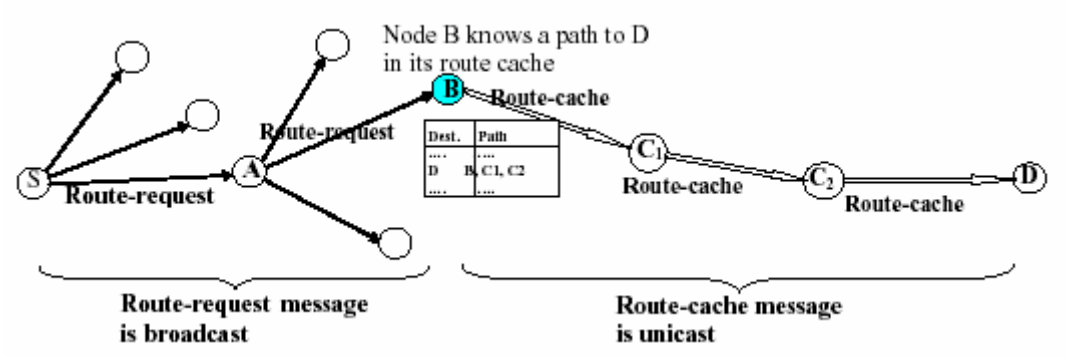



**Figure 3.5 Route-cache message in the LEAR algorithm.**

**Conditional Max-Min Battery Capacity Routing (CMMBCR) Protocol**

As in LEAR, the CMMBCR protocol uses the concept of a threshold to maximize the lifetime of each node and to use the battery fairly. If all nodes in some possible routes between a source-destination pair have larger remaining battery energy than the threshold, the min-power route among those routes is selected. If all possible routes have nodes with lower battery capacity than the threshold, the max-min route is selected. However, unlike LEAR, the threshold value is fixed leading to a simpler design.

The authors of this protocol proposed an interesting performance metric for measuring the energy balance: *expiration sequence*, defined as the sequence of times when mobile nodes exhaust their battery capacity. Traditional metrics for energy balance are variation of remaining battery capacity, ratio of minimum to average remaining battery capacity, and the network lifetime measured as the time when any node exhausts its battery capacity for the first time. Since these metrics provide limited information on energy balance, the expiration sequence gives more accurate information on how fairly energy is expended.

## 3.2.3 Sleep/Power-Down Mode Approach

Unlike the previous two subsections, the sleep/power-down mode approach focuses on inactive time of communication. Since most radio hardware supports a number of low power states, it is desirable to put the radio subsystem into the sleep state or simply turn it off to save energy. Table 4 summarizes hardware low power states and the MAC-level power down modes supported in IEEE 802.11 and Bluetooth wireless LAN protocols as well as typical power consumption values of the devices implementing the protocols. For example, *Lucent*'s *WaveLAN-II* based on *IEEE 802.11 wireless LAN standard* consumes 250 mA and 300 mA when receiving and transmitting, respectively, while consumes only 9 mA in sleep mode.

<table>
<tr><td colspan="3">IEEE 802.11 (Lucent's 802.11 WaveLAN-II supporting 2 Mbps with radio range up to 250 meters )</td><td colspan="2">Bluetooth (Nokia's Bluetooth supporting 768 Kbps with radio range up to 10 – 100 meters )</td></tr>
<tr><td>Hardware state</td><td colspan="3">Mode of Operation (MAC-level)</td><td>Hardware states</td></tr>
<tr><td rowspan="4">Awake</td><td rowspan="3">Active</td><td>Transmit (300 mA)</td><td rowspan="3">Active (40 – 60 mA)</td><td rowspan="4">Connection</td></tr>
<tr><td>Receive (250 mA)</td></tr>
<tr><td>Idle or Listen (230 mA)</td></tr>
<tr><td colspan="2">Power Save</td><td>Sniff Hold</td></tr>
<tr><td rowspan="2">Doze</td><td colspan="2" rowspan="2">Sleep (9 mA)</td><td>Park</td></tr>
<tr><td>Standby (0.55 mA)</td><td>Standby</td></tr>
</table>

**Table 3.4 Power down states and modes**

However, when all the nodes in a MANET sleep and do not listen, packets cannot be delivered to a destination node. One possible solution is to elect a special node, called a *master*, and let it coordinate the communication on behalf of its neighboring slave nodes. Now, slave nodes can safely sleep most of time saving battery energy. Each slave node periodically wakes up and

communicates with the master node to find out if it has data to receive or not but it sleeps again if it is not addressed

In a multihop MANET, more than one master node would be required because a single master cannot cover the entire MANET. Figure 6 shows the master-slave network architecture, where mobile nodes except master nodes can save energy by putting their radio hardware into low power state. The master-slave architecture in Figure 6(a) is based on symmetric power model, where master nodes have the same radio power and thus the same transmission range as slave nodes. On the other hand, Figure 6(b) shows the asymmetric power model, where master nodes have longer transmission range. While this type of hierarchical network architecture has been actively studied for different reasons, such as interference reduction and ease of location management, the problem of selecting master nodes and maintaining the master-slave architecture under dynamic node configurations is still a challenging issue.

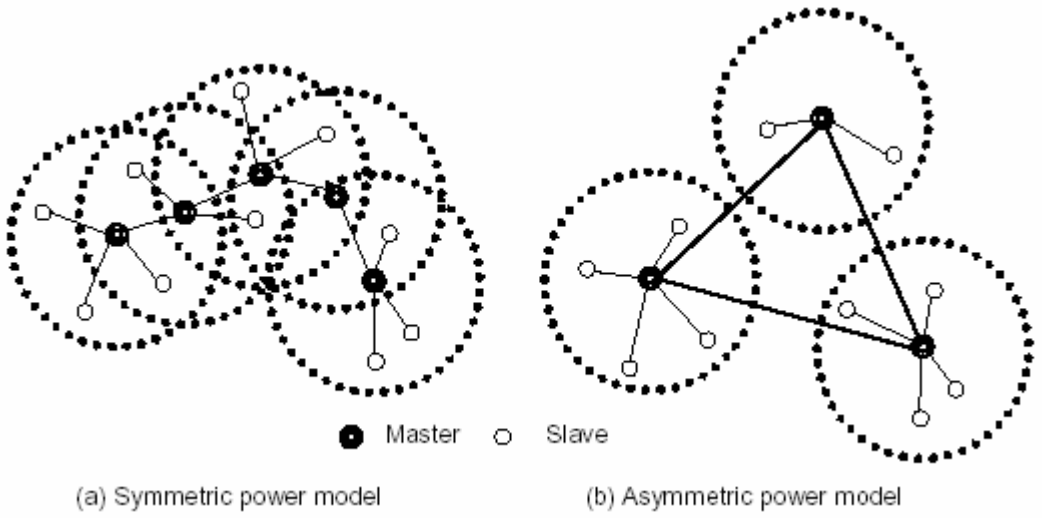



**Figure 3.6: Master-slave MANET architecture**

This subsection introduces three routing algorithms that exploit the radio hardware's low power states. The *SPAN* protocol and the *Geographic Adaptive Fidelity* (GAF) protocol employ the master-slave architecture and put slave nodes in low power states to save energy. Unlike SPAN and GAF, *Prototype Embedded Network* (PEN) protocol practices the *sleep period operation* in an asynchronous way without involving master nodes.

**SPAN Protocol**

To select master nodes in a dynamic configuration, the SPAN protocol employs a distributed master eligibility rule so that each node independently checks if it should become a master or not.

The rule is that if two of its neighbors cannot reach each other either directly or via one or two masters, it should become a master [13]. This is shown in Figure 7 where nodes B and D become masters. If either B or D does not elect itself as a master, node H is eligible (thus, the master selection process is not deterministic). This rule does not yield the minimum number of master nodes but it provides robust connectivity with substantial energy savings. However, the master nodes are easily overloaded. To prevent this and to ensure fairness, each master periodically checks if it should withdraw as a master and gives other neighbor nodes a chance to become a master. Non-master nodes also periodically determine if they should become a master or not based on the master eligibility rule.

Another benefit of the master-slave architecture is that master nodes can play an important role in routing by providing a routing backbone as in Figure 6(a). Control traffic as well as channel contention will also be reduced because the routing backbone helps to avoid the broadcast flooding of route-request messages.

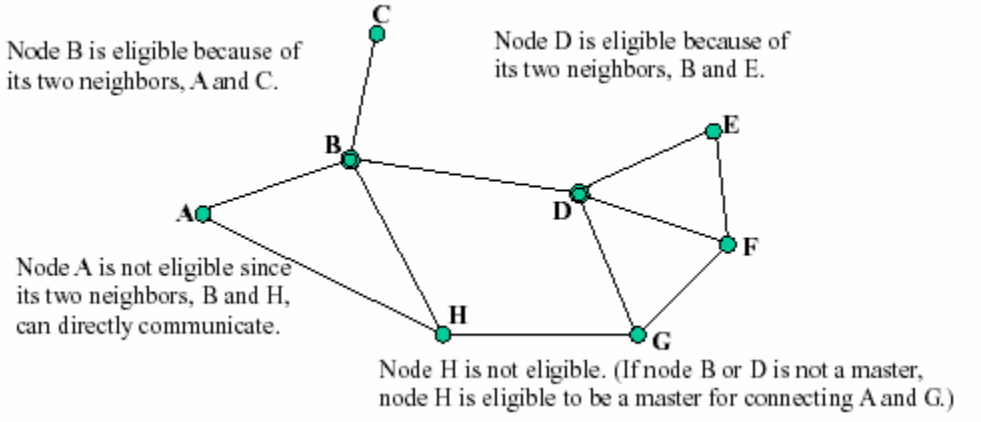



**Figure 3.7 Master eligibility rule in the SPAN protocol.**

**GAF (Geographic Adaptive Fidelity) Protocol**

In GAF protocol, each node uses location information based on GPS to associate itself with a "*virtual grid*" so that the entire area is divided into several square grids, and the node with the highest residual energy within each grid becomes the master of the grid. Other nodes in the same grid can be regarded as redundant with respect to forwarding packets, and thus they can be safely put to sleep without sacrificing the "*routing fidelity*" (or routing efficiency). The slave nodes switch between off and listening with the guarantee that one master node in each grid will stay awake to route packets. For example, nodes 2, 3 and 4 in the virtual grid B in Figure 8 are equivalent in the sense that one of them can forward packets between nodes 1 and 5 while the

other two can sleep to conserve energy. The grid size $r$ can be easily deduced from the relationship between $r$ and the radio range $R$ as $r^2 + (2r)^2 \leq R^2$ or $r \leq R/2.24$.

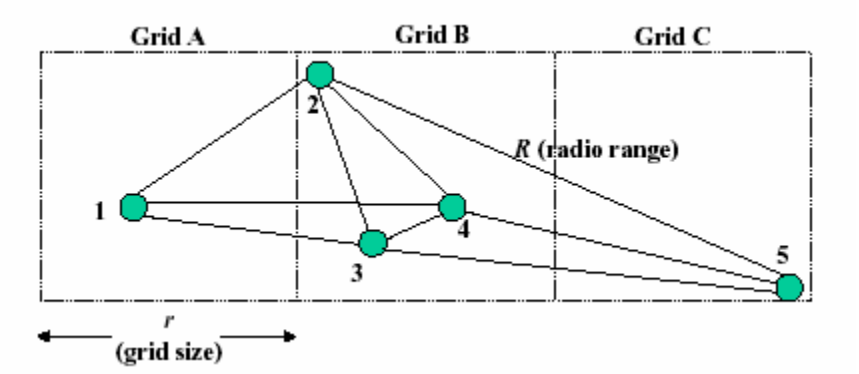



**Figure 3.8 Virtual grid structure in the GAF protocol.**

Master election rule in GAF is as follows. Nodes are in one of three states as shown in Figure 9: *sleeping*, *discovering* and *active*. Initially, a node is in the discovery state and exchanges discovery messages including grid IDs to find other nodes within the same grid. A node becomes a master if it does not hear any other discovery message for a predefined duration *Td*. If more than one node is in the discovery state, one with the longest expected lifetime becomes a master. The master node remains active to handle routing for *Ta*. After *Ta*, the node changes its state to discovery to give an opportunity to other nodes within the same grid to become a master. In scenarios with high mobility, sleeping nodes should wake up earlier to take over the role of a master node, where the sleeping time *Ts* is calculated based on the estimated time the nodes stays within the grid.

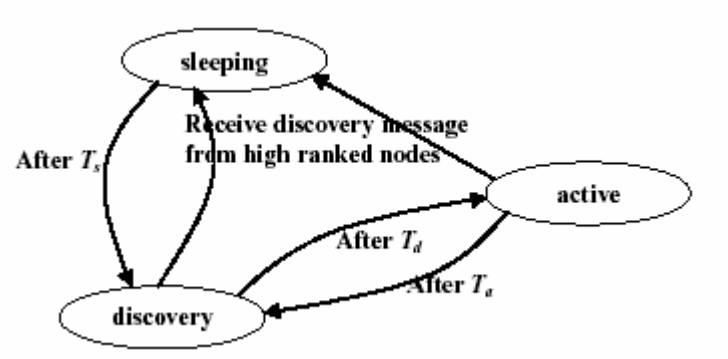



**Figure 3.9 State transition in the GAF protocol.**

**Prototype Embedded Network (PEN) Protocol**

As in SPAN and GAF, the PEN protocol exploits the low duty cycle of communication activities and powers down the radio device when it is idle. However, unlike SPAN and GAF, nodes interact "*asynchronously*" without master nodes and thus, costly master selection procedure as well as the master overloading problem can be avoided. But in order for nodes to communicate without a central coordinator, each node has to periodically wake up, advertises its presence by broadcasting beacons, and listens briefly for any communication request before powering down again. A transmitting source node waits until it hears a beacon signal from the intended receiver or server node. Then, it informs its intention of communication during the listening period of the server and starts the communication. Figure 10 shows those source and server activities along a time chart.

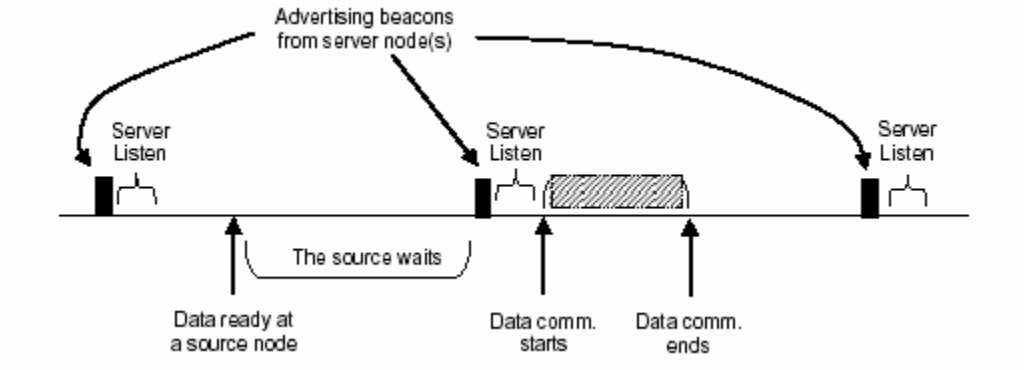



**Figure 3.10 Source and server node activities**

Route discovery and route maintenance procedures are similar to those in AODV, i.e., on-demand route search and routing table exchange between neighbor nodes. Due to its asynchronous operation, the PEN protocol minimizes the amount of active time and thus saves substantial energy. However, the PEN protocol is effective only when the rate of interaction is fairly low. It is thus more suited for applications involving simple command traffic rather than large data traffic.

## 3.3 Choice of Protocol

Current ad hoc routing protocols do not exploit the factors discussed above to conserve energy. The metric that is used is minimum hop in case of the routing protocols discussed in Chapter 2. Our aim is to develop an energy aware protocol that is an extension of an existing on-demand routing protocol. The work in [12] shows that on-demand routing protocols are more efficient in energy consumption as compared to table-based routing protocols. We select Dynamic Source

Routing (DSR) protocol over the AODV protocol for developing the energy aware on-demand routing protocol.

The DSR protocol uses source routing and route caches. This serves as an advantage over AODV since multiple routes are available to a given destination. Combining source routing with promiscous listening, DSR has access to a greater amount of information as compared to AODV that has one entry per destination stored in its route table and does not allow promiscous listening. Since DSR does not have any periodic advertisements, it saves bandwidth and reduces power consumption. AODV requires symmetric links between nodes. DSR on the other hand, utilizes asymmetric links when symmetric links are unavailable. All these reasons prompted us to select the DSR protocol to incorporate energy aware features to develop an energy aware ad hoc routing protocol.

The chapter discussed the design issues concerning the energy aware on demand ad hoc routing protocol. In the subsequent chapter we discuss the design and implementation of the DSR protocol and its energy aware version that we denote as EADSR.

# Chapter 4

# 4. Methodology

## 4.1 Proposed Algorithm to Make DSR Energy Efficient

The interface on a node in a MANET running the modified DSR protocol alternates between being ON (when it can be in transmit, receive or idle state) and being in the sleep state. It starts being ON and remains ON as long as there is network activity (sending, receiving or forwarding data). In the absence of network activity during the period when the interface was ON, the interface is put to the low power sleep state for a period of time. The duration of time for which the node sleeps is initially set to a small value. After each successive no-activity on period the sleep period is linearly increased up to a maximum value to increase energy saving. If network activity resumes during an ON period, the sleep time is reset to its initial value. In addition, if a node, which is in a sleep period, wants to start sending data, it wakes up its network interface immediately. A state diagram for the algorithm is given in Figure 1.

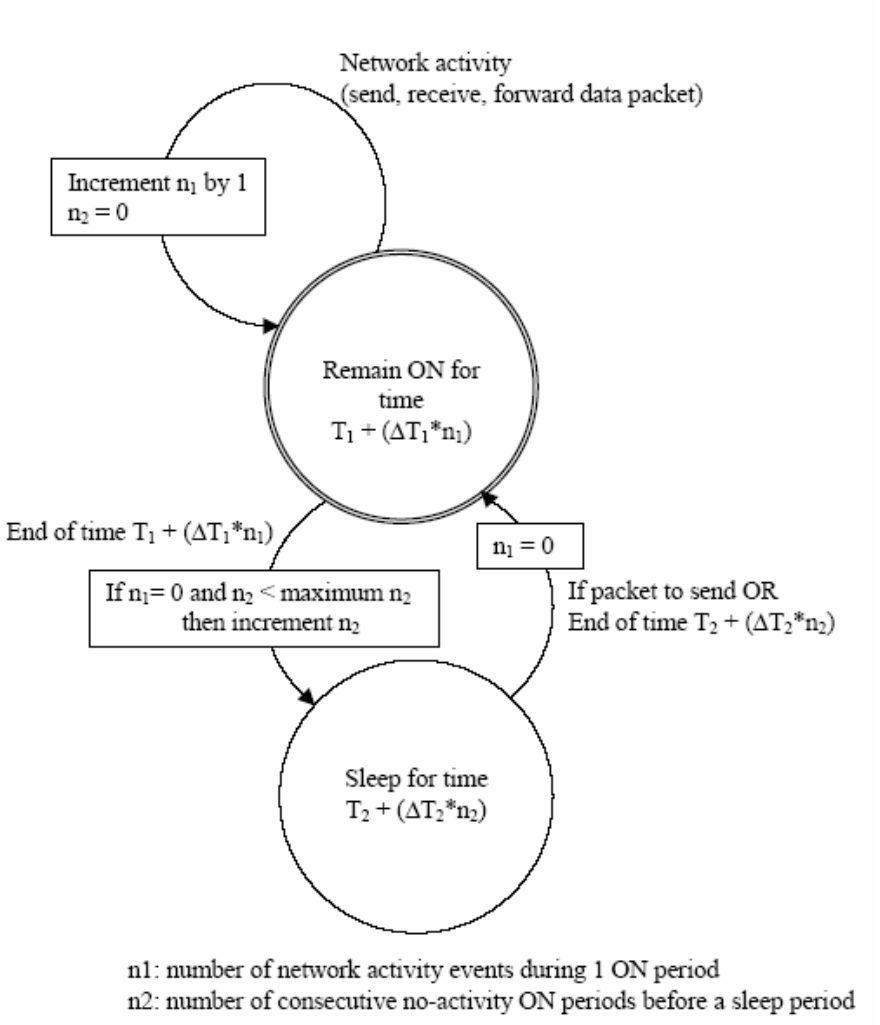



**Figure 4.1 Proposed algorithm for modifying DSR**

In a MANET, due to the presence of random mobility, it is impossible to predict when it is a safe to put an interface to sleep using just the local information about its node. As stated in previous Section, the alternative is to have the position (or information about the remaining battery power) of all the nodes in the system to be available globally to all the nodes. In that scenario, the node would be able to make energy efficient protocol decisions with high degree of accuracy and the performance of the protocol would obviously be better. In this study, an alternative to the requirement for a global positioning system is presented.

A node essentially adapts the time its network interface is ON or asleep according to the current state of the network activity. If there is network activity during the ON period, the node keeps extending the ON period by small amounts proportional to the number of packets received during this period, thereby avoiding transition to the sleep state. The expectation here is that there is a high probability of getting more packets in the near future. Intuitively, this probability

is proportional to the number of packets processed i.e. either sent, received or forwarded so far in this ON period. As stated in previous Section, time and power is spent by a network interface in switching between idle and sleep states. The above approach of prolonging the ON period allows the node to avoid the turnaround time and remain available for network activity for a longer contiguous duration of time.

If there are no packets coming its way, the node put its network interface to sleep. The initial sleep period is kept small to allow the interface to turn itself ON within a short span of time. With each successive no-activity ON period, the node now adapts itself and increases the sleep time of its interface linearly. However, it cannot allow the sleep period to grow too far. In the presence of random mobility, the network activity conditions can change very fast. If the sleep period was allowed to grow too large, the node could potentially miss participating in the routing fabric during the times when its participation was most needed. For example, the network became partitioned or the number of hops required for a bulk of traffic increased significantly due to its non-participation. Capping the maximum sleep period allows the node to react to such network activity changes within an upper time bound and not miss crucial participation completely.

Despite the adaptation performed by the node, it is still possible for a node to put its interface to sleep and miss participating as a router. This is especially true if there is random mobility of the nodes with high speeds. The consequences of putting an interface to sleep have adverse impacts on packet delay and packet loss. It increases the average packet delay, as packets have to wait longer in the queue during their initial route discovery phase. The overall strategy of the algorithm is such that the packet delay is limited to the initial discovery phase. There is an additional caveat here in that the initial route discovery may not result in the most optimal path in terms of the number of hops. This would also result in an increase in packet delay. Once the initial route discovery is over, there would be a high probability that the nodes in the route would have their interfaces up ready to forward data having just recently forwarded the route request and possibly route reply packet. However, the continuous changes in network topology induced by random mobility means that the route discovery needs to be done often leading to a measurable increase in the per-packet delay. In addition, it is also possible for a node to forward a route reply packet during the very end of its ON period. In that case, the extension of its ON period provided in this algorithm may not be enough for it to remain ON when the data packet finally shows up. The node at the previous hop would then have to salvage the packet and generate an error packet back to the source. The above scenarios also lead to an increase in the probability of packet loss in the network as the node at the previous hop may discard the packet altogether if it has no alternate route in its route cache. The probability of packet loss also

increases as the sender at the previous hop may have buffer overflow while it attempts to communicate with a sleeping node.

A MANET already has significant packet loss due to mobility and rapid signal fading characteristics of wireless transmission with increase in distance between the sender and receiver. The proposed algorithm seeks to minimize any additional packet loss by carefully selecting the levels of its factors. The choice of the levels of the factors is presented in Section 7. The choice of metrics for this performance study becomes apparent from the above discussion. The modified protocol was compared with DSR according to the following two metrics:

**Energy consumed per-packet**: The ratio between the total communication cost in the network (equation 1 in Section 4.2) and the total number of packets received by their final destination.

**Delay per-packet**: The ratio between the total delay encountered by the packets received at their final destination in the network and the total number of such packets.

The packet delivery ratio, i.e. the ratio between the number of packets originated by the sources and the number of packets received by their final destination, though important, was not chosen as a metric explicitly. However, the overall goal was to study the performance of modified DSR using the above metrics, with the levels of the factors chosen to keep the packet delivery ratio comparable to that of DSR.

## 4.2 $2^k$ Factorial Design with Replication

A $2^k$ factorial design is used to determine the effect of k factors, each of which has two levels. It helps to sort out the factors in the order of impact. At the beginning of this study, the number of factors and their levels that affect the performance of modified DSR was large. The 2k factorial design was used to reduce the number of factors and to choose those factors that have significant impact. The factors that affect the performance of modified DSR are the following:

**A.** T1: the minimum ON time for a node

**B.** T2: the minimum sleep time for a node

**C.** ΔT1: the amount of time to extend the ON time on each network activity event (send, receive or forward a packet) that occurs during the ON time

**D.** ΔT2: the amount of time to extend the sleep time on each consecutive no activity ON period before this sleep period

**E.** Pause time: the amount of time a node pauses before moving on to another destination. A 0s pause time means constant mobility for the 50 nodes in the system and a 900s pause time means stationary nodes.

**F.** Maximum T2: the maximum amount of time a node can sleep during its sleep period.

**G.** Constant Bit Rate (CBR): the sending rate (in number of packets per second) at each of the 20 source nodes.

For the initial study, maximum T2 was fixed at twice the value of T2 and CBR was fixed at 4. The two levels for the other five factors (k = 5) were the following:

A. T1: 30ms and 300ms
B. T2: 30ms and 300ms
C. ΔT1: 0.01ms and 0.1ms
D. ΔT2: 0.01ms and 0.1ms
E. Pause time: 0s and 900s

The number of replications done was three for the $2^k$ factorial experimental design. In each replication, a different movement pattern of the 50 nodes was used. These patterns are stored as scenario files and each run of the Ns-2 simulator accepts a scenario file as input. A scenario file describes the motion of each node, together with the time at which each change in motion at a node is going to occur. The entire table of results of the 2k factorial design is large and has been omitted for brevity. A summary of results for the performance metrics–per-packet energy consumed and per packet delay is presented in Tables 2 and 3 respectively. The summary lists those factors that explain a high percentage of variation and are thus the most important of the factors.

As stated in the previous section, packet delivery ratio was not listed as a metric explicitly. However, the goal of the study was to keep the packet delivery ratio close to that in DSR. A couple of observations emerged from analyzing the results of the initial experiments conducted for the $2^k$ factorial design. These observations helped in devising the overall strategy to be followed to fix the levels of factors in order to achieve the above goal. The two observations are:

a. The factors that made the most impact were T2 and ΔT2 (factors B and D in the Tables 2 and 3) and their combination. As would be expected, the maximum energy saving is when T2 and ΔT2 are at high levels and T1 and ΔT1 are at low levels. However, the packet delivery ratio suffers an adverse impact in this situation.

b. The packet delivery ratio and the per-packet delay improved significantly when T1 and ΔT1 are at high levels and T2 and ΔT2 are at low levels.

The above observations led to the strategy to start the experiment with a large value of T1 and a small value of T2. In addition, T2 should be allowed to grow to a larger maximum value. This would give continuous ON time to nodes that are involved in network activity and allow nodes that are not involved in network activity to maximize their sleep time in order to conserve energy. Therefore, the list of factors and their levels used for the performance comparison of modified DSR with that of DSR presented in the next section were:

A. Maximum T2: 0.2s to 0.55s (this factor combined the effect of the factors T2 and ΔT2– the factors that were found to have the most influence on the variation of the metrics)
B. CBR: 2, 3, 4, 5 packets per second (this factor validated the correctness of the modified protocol to handle varying amounts of traffic)
C. Pause time: 0s and 900s (this factor again was used to validate the correctness of the modified protocol to handle constant mobility as well as no mobility)

Because of the $2^k$ analysis, some of the initial factors were now considered as workload parameters. T1 was set at a large initial value of 0.32s, T2 was fixed at a small initial value of 0.01s, ΔT2 was set at 0.05s and ΔT1 was set at 0.1s. The number of replications (scenario files) used for the presentation and analysis in the next section was 10.

## 4.3 Simulation Setup and System Parameters

The results in this study are based on simulations done using the Ns-2 simulator. Ns-2 is a discrete event simulator that was developed by the University of California at Berkeley and the VINT project for simulating the behavior of network and transport layer protocols in a complex network topology. It has been extensively enhanced by the Monarch project at the Carnegie Mellon University for use in simulating MANETs. At the physical layer, the radio model in Ns-2 supports propagation delay, a two-ray ground reflection model and collision and capture effects. At the link layer, the simulator implements the complete IEEE 802.11 standard Medium Access Control (MAC) protocol.

I made use of ns-2, which has support for simulating an ad-hoc wireless environment. Along with ns-2, I also made use of two scenario-generation utilities available from the Monarch web page namely setdest and the tcl script cbrgen.tcl. setdest was used to generate the movement of the mobile nodes and cbrgen.tcl was used to generate the traffic patterns. Initially I used the generated scenarios for the simulations. But to double check the results I was getting I decided to use the scenarios used by the Monarch Group for their simulations.

I modified a file for wireless simulations provided with ns-2 distribution for our project. The simulation generated huge trace files (of the order of 70 MB) and I parsed these files to obtain the performance metrics. I determined the following metrics from the simulations:

- **Energy consumed per-packet**: The ratio between the total communication cost in the network and the total number of packets received by their final destination.
- **Delay per-packet**: The ratio between the total delay encountered by the packets received at their final destination in the network and the total number of such packets

For comparing the performance of the two protocols, the scenarios used in the simulations were similar to those used by the Monarch project. The following is a discussion of the simulation setup.

The network model consists of 50 nodes in a 1500 x 300 meter flat rectangular area. All nodes communicate with identical wireless radios that are modeled after the commercially available IEEE 802.11 WaveLAN PC card. The cards have a bandwidth of 2Mbps and a transmission radius of 250m. In each experiment run, the nodes move around in the rectangular area for 900s of simulated time. The mobility of the nodes is based on the random waypoint mobility model. In the model, each node picks a random destination and speed (between 0 and 20 m/s; average speed is 10 m/s) in the rectangular area and then moves to the destination in a straight line. Once the node reaches its destination, it pauses for a pause time, picks another destination and speed and moves on. The workload parameters consist of 20 randomly chosen source destination pairs of nodes that provide the traffic load. Each source sends a constant bit rate (CBR) stream of 64 byte packets using UDP. The small size of the data packets is used to factor out the congestive effects of using a large packet size. A node could act as the source or destination for more than one stream. The twenty connections start at times uniformly distributed between 0s and 180s. As explained in previous Section, the transmission of each unicast packet is preceded by an exchange of RTS/CTS control packets. The network interfaces on the 50 nodes operate in promiscuous mode enabling the use of all optimizations in the DSR protocol.

To have a fair comparison of the capabilities and the weaknesses of the modified DSR and Original DSR we tested them with identical movement patterns and workloads. Each run of the simulator accepts as input a *scenario file* that describes the exact motion of each node and another which details the traffic flows in the network during the. I have submitted a sample traffic pattern file (CBR-50-20-4-64) and a movement scenario (scen-1500x300-50-30-20-1) in my submission. As mentioned earlier I used pattern and scenario generated by the Monarch group and available at their website for download. I then ran the original DSR routing protocols and the modified version against each of these scenario files.

The simulations were carried out on an AMD Duron – 1.1 GHz system running Suse Linux 9.0. Each simulation of 900s (simulation time) took approximately 6 minutes to complete on this machine. The simulation generated trace files that contained a list of all major events such as packet transmission, receipt and drops during the simulation. The events for the both AGENT module (the data source) and the ROUTER module (the module that implements the routing algorithm) were traced. The extraction of the results from these trace files is described later.

## 4.4 Movement Scenarios

The movement scenario files we used for each simulation were characterized by a *pause time*. Each node begins the simulation by remaining motionless for *pause time* seconds. It then selects a random destination in the 1500m*300m space and proceeds to that destination at a speed of 20m/s. Upon reaching the destination, the node again stays stationary for *pause time* seconds, selects another destination and moves there as previously described, repeating this behavior for the duration of the simulation. Each simulation ran for 900 seconds of simulated time. We ran our simulations with movement patterns generated for 8 different pause times: 0, 30, 60, 120, 300, 600 and 900 seconds. A pause time of 0 seconds corresponds to continuous motion, and a pause time of 900 corresponds to almost no motion. We used a set of five movement scenario files for each value of pause time to get to improve the accuracy of the results and to smooth out spikes due to extremely favorable or unfavorable movement scenarios. The movement scenarios were named according to the parameters used to generate them e.g. for the file scen-1500x300-50-30-20-1 there are 50 nodes moving about in an area of 1500 * 300 space with an average pause time 30 seconds and a maximum speed of 20 m/s.

## 4.5 Communication Models

As the goal of our simulation was to compare the performance of the two routing protocol, we chose our traffic sources to be constant bit rate (CBR) sources. During the course of the simulation 20 connections were setup between the nodes in the network. We used a sending rate of 2, 3, 4 and 5 packets of 64 bytes each per second. TCP was not used as a source because TCP implements timeouts and retransmissions and also adjusts its sending rate according to the perceived network. As a result, both the time at which each data packet is originated by its sender and the position of the node when sending the packet might differ between the protocols, preventing a direct comparison between them. For all our simulations we used the same traffic pattern file (CBR-50-20-4-64) for the simulations. The different values in its name represent the number of nodes, the number of connections, the number of packets sent per second and the size of the packets.

## 4.6 Calculation of Results

We used the "grep" utility in Linux to extract the results from the trace files that were generated. Each event is represented by one line in the trace file and ns-2 includes the information about the type of event and the module that is logging the event in each line e.g. drops start with a capital 'D' and the events logged by the ROUTER module have RTR So. We searched the trace files for these patterns to determine the number of events of each type. Since the set of operations had to be conducted for each trace file we wrote shell scripts to automate this process.

## 4.7 Design Rationale

In this section we talk about the different choices we made for this project and our motivation behind them. First of all we decided to use a simulator for out performance study because a practical implementation of an ad hoc wireless network was obviously not feasible. We chose ns-2 as the simulator primarily because the protocols had already been implemented before and thus provided us with a ready platform for our simulations. Another reason for choosing ns-2 was it's widespread use in the academic community and the comprehensive manuals and tutorials that are freely available for ns-2.

After choosing ns-2 we decided to concentrate on the ad hoc network protocols that have been implemented for it namely DSDV, DSR, AODV and TORA. The first sets of simulations that we carried out were using the original DSR protocol. For the rest of the simulations we carried out the simulation for the modified DSR.

# Chapter 5

# 5. Network Simulator 2

As mentioned earlier in the text, Network Simulator 2 is used as the simulation tool in this project. NS was chosen as the simulator partly because of the range of features it provides and partly because it has an open source code that can be modified and extended. In this project, the Dynamic Source Routing has been modified in NS 2 so that the modified DSR becomes more energy efficient than the original DSR.

There are several different versions of NS and at the current time the latest version is ns-2.26. While ns-2.27 is under development. The latest version of ns has been used in this study.

This Chapter describes the simulation environment in Section 5.1 and the application used for animation in Section 5.2.

## 5.1 Network Simulator (NS)

Network Simulator (NS) is an object-oriented, discrete event simulator for networking research. NS provides substantial support for simulation of TCP, routing and multicast protocols over wired and wireless networks. The simulator is a result of an on-going effort of research and development. Even though there is a considerable confidence in NS, it is not a polished and finished product. Bugs are being discovered and corrected continuously.

NS is written in C++, with an OTcl interpreter as a command and configuration interface. The C++ part, which is fast to run but slower to change, is used for detailed protocol implementation. The OTcl part, on the other hand, which runs much slower but can be changed very quickly, is used for simulation configuration. One of the advantages of this split-language programming approach is that it allows for fast generation of large scenarios. To simply use the simulator it is sufficient to know OTcl. On the other hand one disadvantage is that modifying and extending the simulator requires programming and debugging in both languages simultaneously.

### 5.1.1 Marc Greis' Tutorial

The very first thing to do for a new user of NS, is to read Marc Greis' tutorial. There is a link to this tutorial on the web page of NS which can be found at www.isi.edu/nsnam/ns/. The purpose of this tutorial is to make it easier for new NS users to use NS and NAM, to create their own simulation scenarios for these tools and to eventually add new functionality to NS.

### 5.1.2 NS by Example

On the web page of NS, there is a link to another tutorial for NS. This tutorial has been written by Jae Chung and Mark Claypool and its purpose is to give new users some basic idea of how the simulator works, how to setup simulation network codes, how to create new network components and so on. In particular, it explains the linkage between the two languages used in NS, namely C++ and OTcl. One can find some very good examples and brief explanations in this tutorial, which is the second tutorial to study after reading Marc Greis' tutorial.

### 5.1.3 NS Manual, NS Search, and NS Mailing List

In the NS Manual one can find the answer to many questions. A link to this Manual can be found on the web page of NS. However, if no answer can be found in the Manual the NS mailing list archives should be searched. The archive keeps all previous emails sent to the ns-users mailing list. The ns-users mailing list should be used if an answer still (after looking in the Manual and searching the archives) has not been found. Everyone that has subscribed will receive this email and will hopefully reply.

## 5.2 Network Animator (NAM)

Network Animator (NAM) is an animation tool for viewing network simulation traces and real world packet traces. It supports topology layout, packet level animation and various data inspection tools.

Before starting to use NAM, a trace file needs to be created. This trace file is usually generated by NS. It contains topology information. E.g. nodes and links, as well as packet traces. During a

simulation, the user can produce topology configurations, layout information and packet traces using tracing events in NS.

Once the trace file is generated, NAM can be used to animate it. Upon startup, NAM will read the trace file, create topology, pop up a window, do layout if necessary and then pause at time o. Through its user interface, NAM provides control over many aspects of animation.

Although the NAM software contains bugs, as do the NS software, it works fine most of the times and causes only little trouble. NAM is an excellent first step to check that the scenario works as expected. NS and NAM can also be used together for educational purpose and to easily demonstrate different networking issues.

# Chapter 6

# 6. Simulation Results and Conclusions

## 6.1 Comparison of the Original DSR with the Modified DSR

Figures 1 and 2 show the performance of the modified protocol and that of DSR under different constant bit rates (traffic loads) of 20 sources with a pause time of 0s. The graphs for the modified protocol also show the 90% confidence interval error bars. Figure 1 shows that the maximum percentage of energy saved in each of the traffic load cases corresponds to the maximum sleep time of 0.55s. It is 25% for the CBR 2 traffic load, 20% for both CBR 3 and 4 and about 21% for CBR 5 traffic load. The error bars for the energy plots for DSR in Figure 1 are very narrow and are therefore not shown.

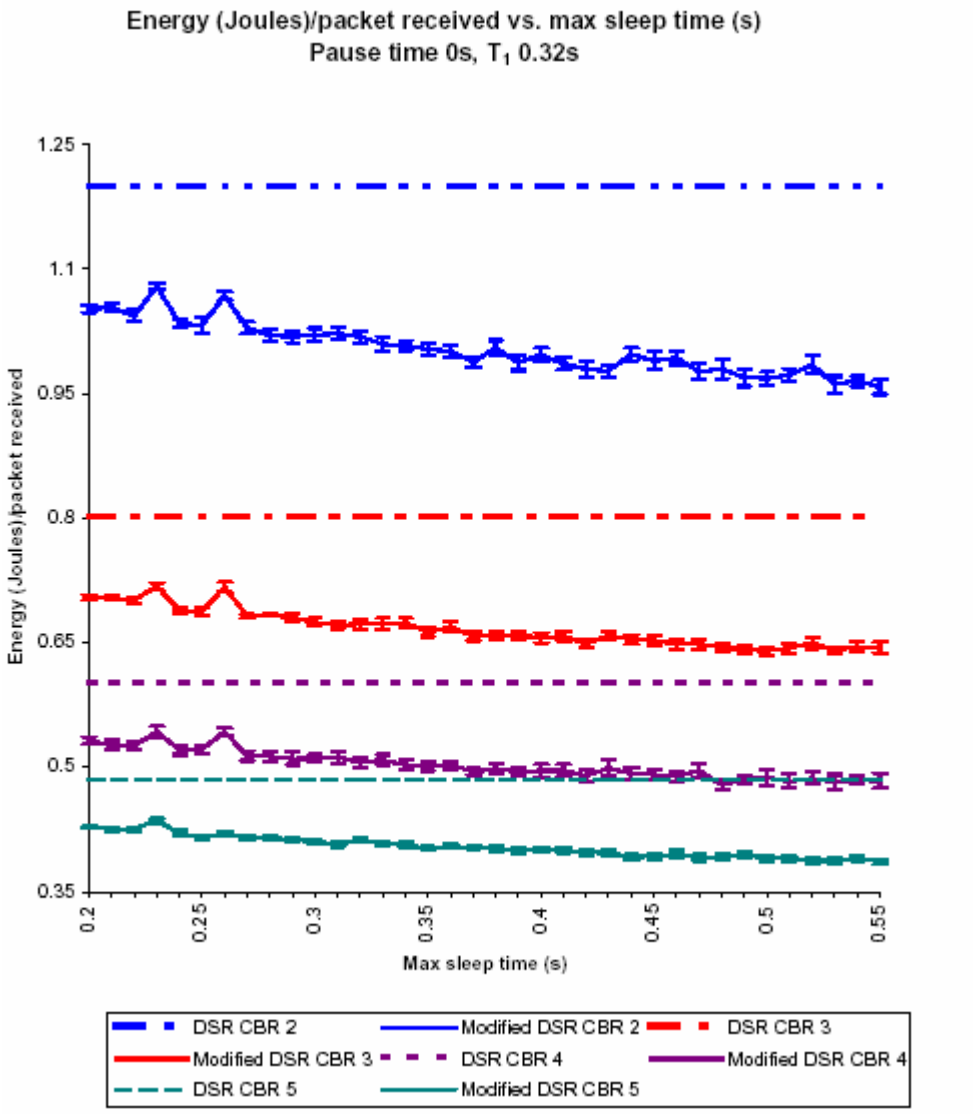



**Figure 6.1: Energy consumption per-packet for the pause time of 0s**

As explained in Section 4, some amount of increase in delay is inevitable. Figure 2 shows the per-packet delay for all traffic loads. The increase in per-packet delay is more than 3ms for CBR 2, it is less than 3ms for CBR 3, it is about 2ms for CBR 4 and it is less than 2ms for CBR 5. The error bars for DSR have not been shown in Figure 2 to limit the amount of data points shown in an already dense graph. Figure 4 shows the per-packet delay plots along with the 90% confidence interval error bars for modified DSR and DSR when CBR is 4. Other traffic load cases have plots similar to that in Figure 4.

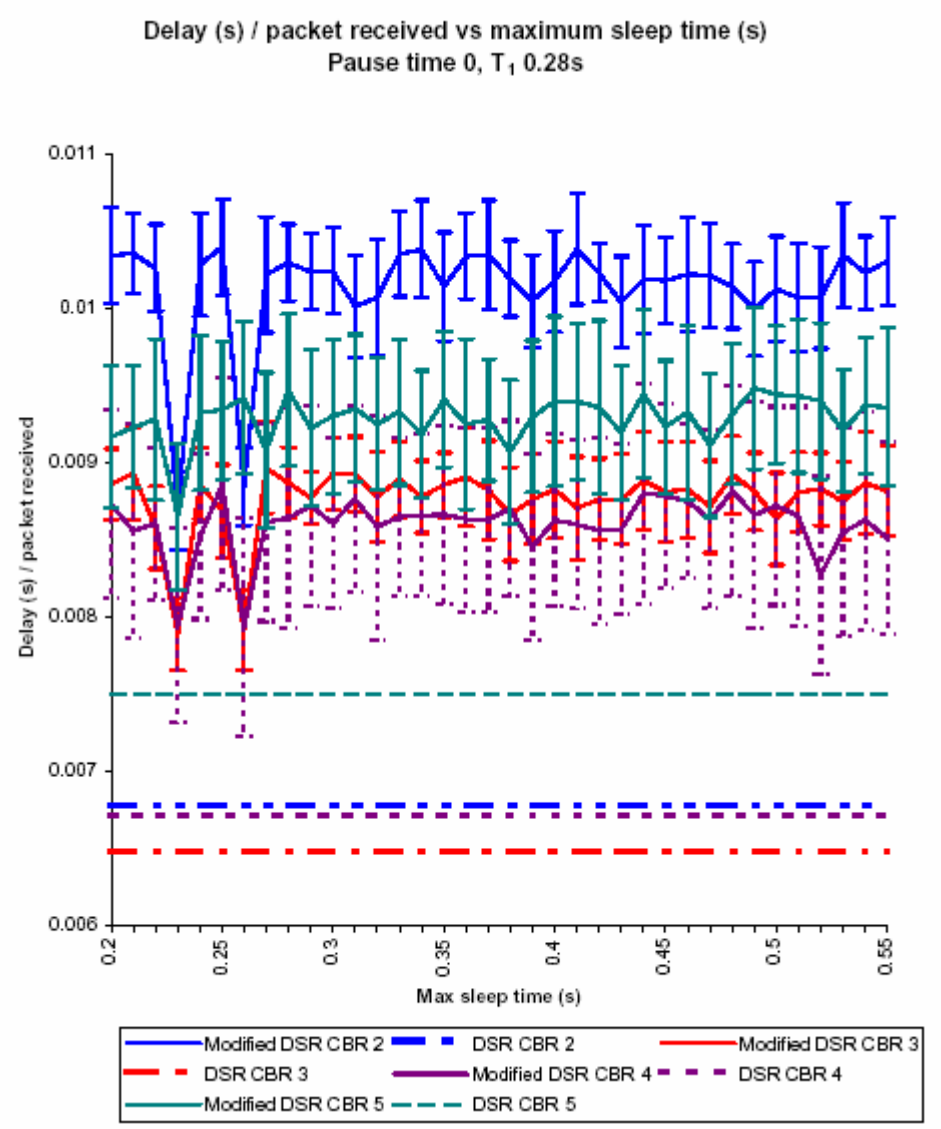



**Figure 6.2: Delay per packet for the pause time of 0**

The relative decrease in the per-packet delay as the traffic load increases supports the conclusion regarding the correct behavior of the modified protocol. As the amount of traffic increases, the nodes in the region of network activity have a higher probability to stay ON to handle the activity than when the amount of traffic in the network is low. There is no visible trend in the per-packet delay for any of the traffic loads. This validates the strategy in the modified protocol where the nodes are able to adjust their sleep times according to the network activity. As the maximum sleep time is increased, the per-packet energy consumption decreases as nodes that are not involved in network activity get to sleep for longer time. However, the nodes that need to be involved are ON with almost the same probability in both the low and the high maximum sleep time cases. This is shown in Figure 5 that plots the percentage of total time spent in each of the 4 energy consumption states by each of the 50 nodes for a simulation run. The top plot in Figure 5

shows the distribution when the maximum sleep time is 0.2s while the bottom plot shows the distribution when it is 0.55s.

Figure 3 show that some of the nodes spend very little time in the sleep state. For example, consider nodes numbered 7 and 8. Node 7 is a sender and the transmission of the CBR stream from node 7 to 8 starts after only 6s of simulation time. However, as is shown in Figure 3, the percentage of time spent in the sleep states by node7 and 8 is not identical. Node 7 cannot sleep at all after 6s as it sends a packet out about every 0.25s (CBR is 4) while T1 is 0.32s. On the other hand, node 8 goes through an interval of time when it is disconnected from node 7 during which it spends some time in the sleep state. This was verified by visually looking at the generated trace files using the Network Animation (NAM) tool in Ns-2.

Apart from nodes numbered 1, 2, 11, 12 and 13 in Figure 3, the other nodes with numbers in the range of 3 through 24 are either senders or receivers and get to spend less time in the sleep state. As previously stated, the start time for the senders are staggered between 0s and 180s. Therefore, not all sender nodes have the same pattern as node 7. Most of them do get to be in the sleep state before they start sending data that keeps them ON throughout the remainder of the simulation run. As would be expected, other nodes in the system also remain ON to participate in the routing fabric.

The different maximum sleep times in the top and bottom plots in Figure 3 cause different nodes to accept the role of forwarding traffic. Node 2 in the top plot and node 36 in the bottom plot are one such example of nodes that stay ON to forward traffic. The per-packet delay and the packet delivery ratio in both the simulation runs are almost identical. This further validates the ability of the modified protocol to support traffic even when the interfaces on some of the nodes are put to sleep state to conserve energy. The bottom plot in Figure 4 also demonstrates the ability of a few of the nodes using the modified protocol to save a lot of energy. The nodes numbered 26, 31 and 37 in the bottom plot, not involved in any network activity, spend as much as 50% of the total time with their interfaces in the sleep state.

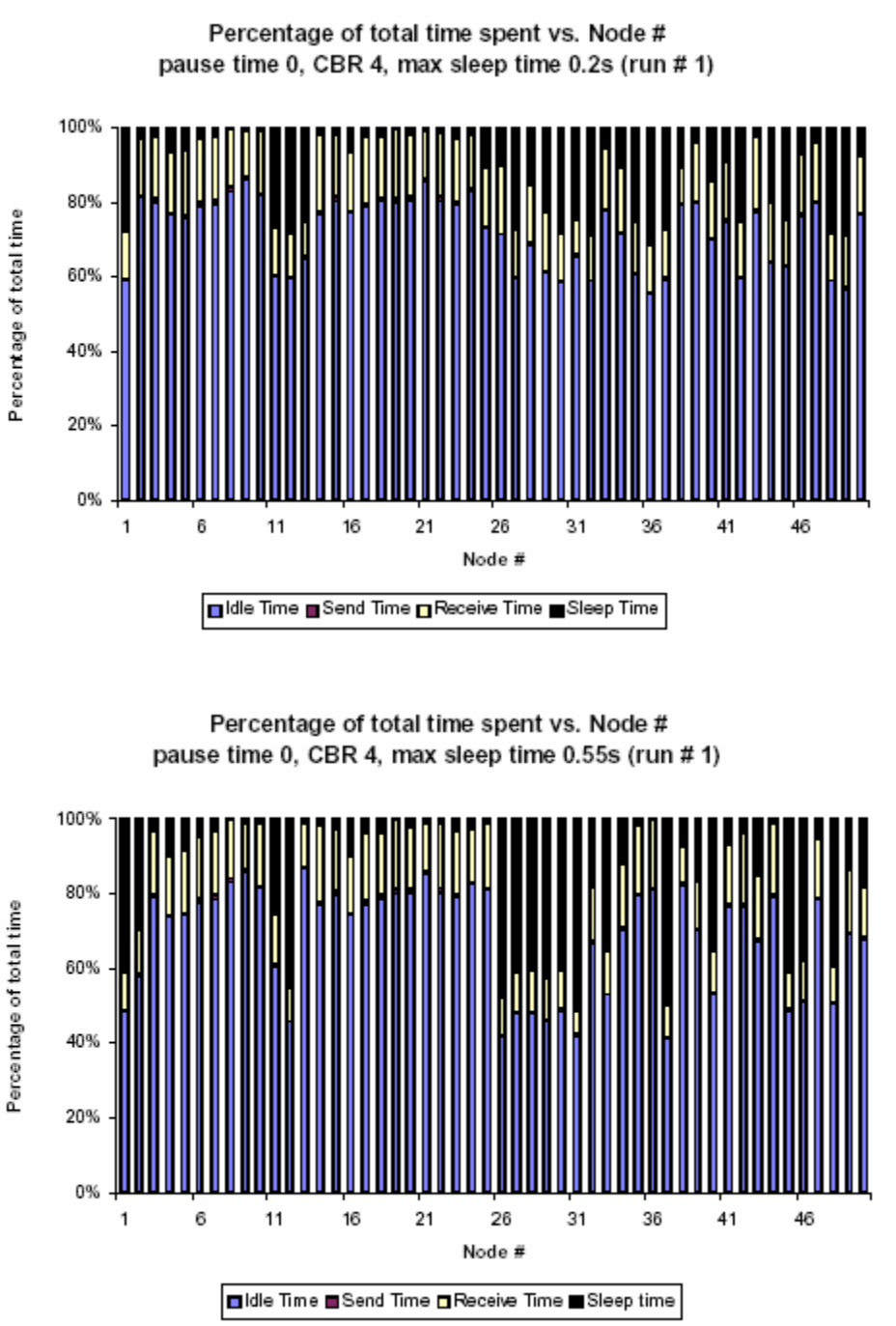



**Figure 6.3 Percentage of time spent in each of the 4 states by the 50 nodes**

Figures 5 and 6 show the performance of the modified protocol to that of DSR under constant bit rate of 4 packets/s of 20 sources with a pause time of 900s. Figure 5 shows that the maximum energy saved in this case is about 29% when the maximum sleep time is 0.54s. As is expected, the energy saving is higher when there is no mobility as the nodes that are never in the network activity region are able to sleep the maximum possible amount of time. Figure 6 shows that the

average per-packet delay is less than 2ms. This is unusually high. However, closer analysis of the trace files generated by the modified protocol shows that most of the delay is caused in the initial 200s of a simulation run. As sender nodes start sending packets (in a staggered manner between 0s and 180s), it takes some time before the initial packets reach their destination. If the initial 200s are removed from the analysis, the per-packet delay is almost identical (less than a fraction of 1ms) for both modified DSR and DSR. The remaining traffic load cases (CBR of 2, 3 and 5) were not considered for pause time of 900s. Their results would follow the same trend as that of the results when CBR is 4 as was shown very clearly in Figure 1 for pause time of 0s.

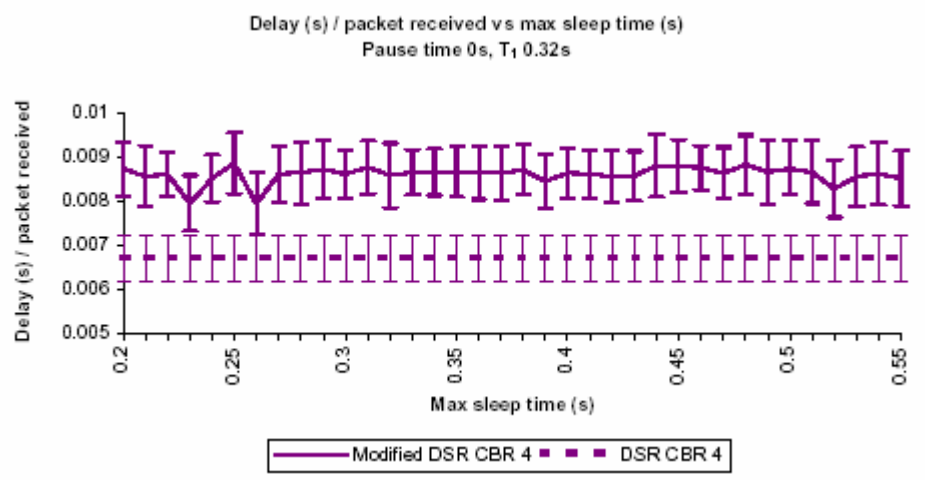



**Figure 6.4 Delay per packet for the pause time of 0s and CBR of 4**

Two other details can be noted from the graphs presented in this report. One is that the per-packet energy consumption is dominated by the energy consumed by the nodes in the idle state as shown in Figure 3. The idle portion of the energy consumption is distributed among a higher number of packets as the traffic load is increased resulting in the lowering of the per-packet energy consumption (as shown in Figure 1). The other point is that the percentage of time spent by a node in the send state is a very small percentage of the total time. Similar results showing the high cost of idle state power consumption and the fact that the number of packets sent is quite small in comparison with the total traffic were also reported in [4].

Figure 7 shows the comparison between the maximum possible energy saving calculated from applying equation (10) of Section 2.8.2 and the actual energy saved for one simulation run when CBR is 4. To reiterate, the calculated maximum value is the ratio between the average sleep time of all the nodes in the system and the total simulation time. The difference between the calculated maximum value and the actual value is because in deriving equation (10), it is assumed that the sleep time leads to a reduction in idle time only. However, in actuality, the presence of sleep time

for nodes, while decreasing the idle time energy consumption, also leads to an increase in the send time for the nodes negating some of the energy saving. This was confirmed by analyzing the data recorded from the simulation run (the actual data is not presented in this report for brevity). This behavior is expected as the presence of nodes with interfaces in the sleep state increases the total time required for sending data. This can intuitively be attributed to the use of longer routes by the data traffic (in terms of number of hops) as explained in Section 2.8.2.

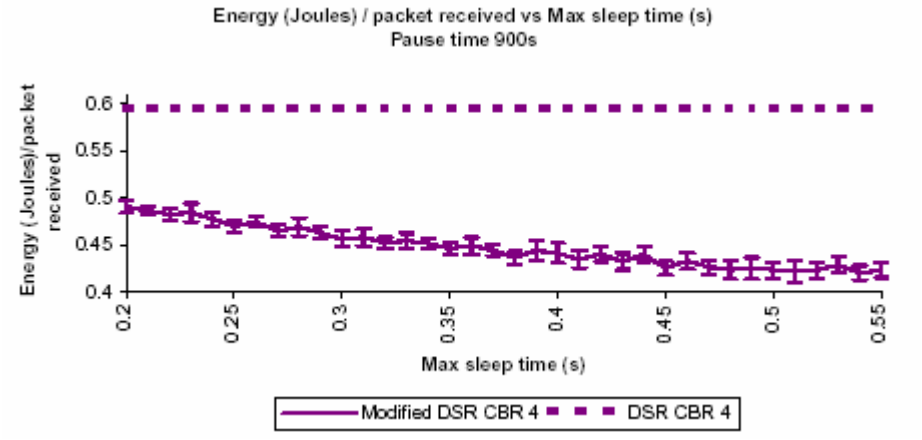



**Figure 6.5 Energy Consumption per-packet for pause time of 900s and CBR of 4**

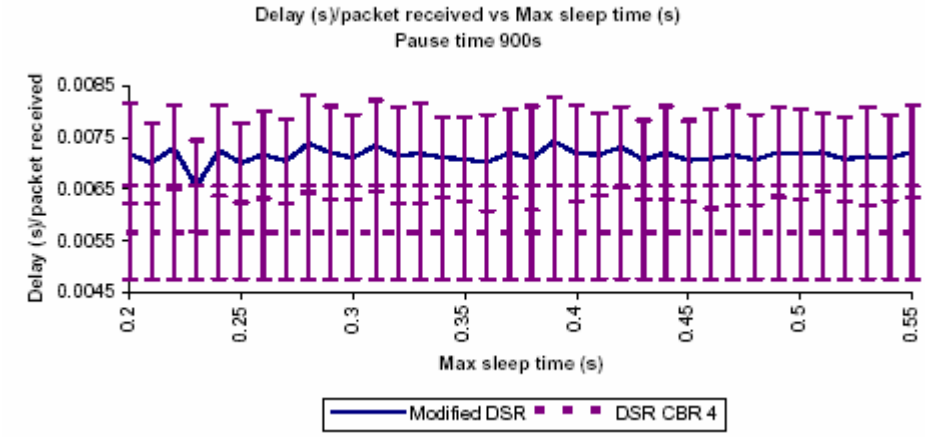



**Figure 6.6 Delay per-packet for pause time of 900s and CBR of 4**

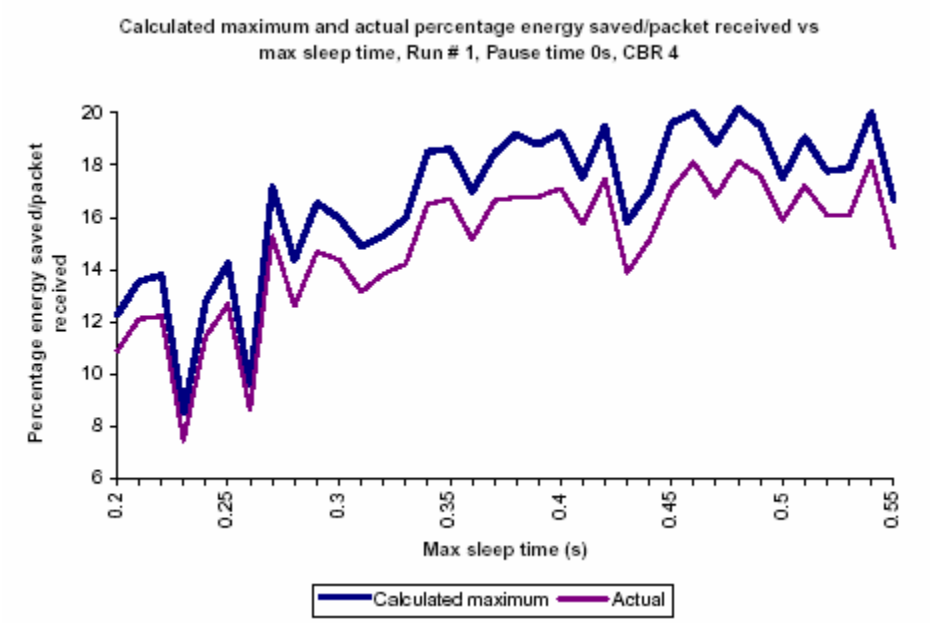



**Figure 6.7 Calculated and Actual Energy saved per-packet**

## 6.2 Conclusions

Reducing power consumption in ad hoc networks has received increased attention among researchers in recent years. The idle mode power consumption in the nodes is significant in MANETs. In this thesis, modifications were made to DSR to reduce the power consumption in the nodes operating in a MANET. The simulations presented here show the effectiveness of the modified protocol to save energy and yet have a minimal adverse impact on packet delay in the network. By selecting the appropriate levels of the factors affecting the modified protocol, the packet delivery ratio achieved is comparable to that of DSR.

The advantage in this approach is that the nodes require no additional information (for example, the positions of all the nodes in the network) required in other protocols designed for reducing power consumption in MANETs. The nodes also do very little extra computation to participate in the modified protocol when compared to some of the other approaches. While a number of factors were considered for this study, some others like wireless transmission speeds and ranges, effective node density, type of sources (whether UDP or TCP) and data packet size were treated as system and workload parameters. These will clearly have an impact on the overall energy utilization and they must be considered in a future study.

# Appendix A
# Implementation of DSR protocol in NS2

The following sections describe some details about implementation of the Dynamic Source Routing protocol in MANET. The most important modifications and extensions are presented in this Appendix.

## A.1 Modifying the Energy Model Implementation in NS2

The Energy model related files in the distribution are energy-model.{h,cc}. The main modifications made to the energy-model.h will be described in the section below.

### A.1.1. Modifications in Energy-model.h

```
#define MODIFY_DEREJE
class EnergyModel : public TclObject {
public:
        EnergyModel(MobileNode* n, double energy, double l1, double l2) :
                energy_(energy), initialenergy_(energy),
        #ifdef MODIFY_DEREJE
                , last_wakeup_time(0.0)
                double get_last_wakeup_time() { return last_wakeup_time; }
protected:
          #ifdef MODIFY_DEREJE
                double last_wakeup_time;
          #endif
};
```

### A.1.1. Modifications in Energy-model.cc

```
void EnergyModel::DecrTxEnergy(double txtime, double P_tx)
{
```

```
	double dEng = P_tx * txtime;
#ifdef MODIFY_DEREJE
	add_sndtime(txtime);
#endif // MODIFY_DEREJE
	if (energy_ <= dEng)
		energy_ = 0.0;
	else
		energy_ = energy_ - dEng;
	if (energy_ <= 0.0)
		God::instance()->ComputeRoute();
}

void EnergyModel::DecrRcvEnergy(double rcvtime, double P_rcv)
{
	double dEng = P_rcv * rcvtime;
#ifdef MODIFY_DEREJE
	add_rcvtime(rcvtime);
#endif // MODIFY_DEREJE
	if (energy_ <= dEng)
		energy_ = 0.0;
	else
		energy_ = energy_ - dEng;
	if (energy_ <= 0.0)
		God::instance()->ComputeRoute();
}

void EnergyModel::set_node_sleep(int status)
{
	Tcl& tcl=Tcl::instance();
	//static float last_time_gosleep;
	// status = 1 to set node into sleep mode
	// status = 0 to put node back to idle mode.
	// time in the sleep mode should be used as credit to idle
	// time energy consumption
	if (status) {
		last_time_gosleep = Scheduler::instance().clock();
```

```
#ifdef DEBUG_DEREJE
                printf("EM: put node (%d) into sleep at %f\n",
                  node_->nodeid(), last_time_gosleep);
#endif // DEBUG_DEREJE
                //printf("id=%d : put node into sleep at %f\n",
                // address_,last_time_gosleep);
                sleep_mode_ = status;
                if (node_->exist_namchan())
                        tcl.evalf("%s add-mark m1 blue hexagon",node_->name());
        } else {
#ifdef MODIFY_DEREJE
                last_wakeup_time = Scheduler::instance().clock();
#endif // MODIFY_DEREJE
#ifdef DEBUG_DEREJE
                printf("EM: last wakeup time for node(%d) : %f\n",
                  node_->nodeid(), last_wakeup_time);
#endif // DEBUG_DEREJE
                sleep_mode_ = status;
                if (node_->exist_namchan())
                        tcl.evalf("%s delete-mark m1", node_->name());
                //printf("id= %d last_time_sleep = %f\n",
                // address_, last_time_gosleep);
                if (last_time_gosleep) {
                        total_sleeptime_ += Scheduler::instance().clock() -
                                last_time_gosleep;
                        last_time_gosleep = 0;
                }
        }
}
```

## A.2 Modifying the DSR agent and related files implementation in NS2

The DSR and related files that are modified in the distribution of Network Simulators are

- loss-monitor.cc
- loss-monitor.h
- udp.cc
- udp.h
- mobilenode.cc
- mobilenode.h
- dsragent.cc
- dsragent.h

### A.2.1. Modifications in dsragent.h

```
#ifdef MODIFY_DEREJE
class PutNodeToSleepTimer;
class WakeupNodeTimer : public TimerHandler {
public:
  WakeupNodeTimer(DSRAgent* a, PutNodeToSleepTimer* s) : TimerHandler() { a_ = a; s_ = s;}
  void expire(Event *e);
protected:
  DSRAgent *a_;
  PutNodeToSleepTimer* s_;
};

class PutNodeToSleepTimer : public TimerHandler {
public:
  PutNodeToSleepTimer(DSRAgent* a, WakeupNodeTimer* w) : TimerHandler() { a_ = a; w_ =
w;}
  void expire(Event *e);
  double timerToExpireAt;
protected:
  DSRAgent *a_;
  WakeupNodeTimer* w_;
};

#endif // MODIFY_DEREJE

class DSRAgent : public Tap, public Agent {
        Private:
                #ifdef MODIFY_DEREJE
```

```
                PutNodeToSleepTimer putnodetosleep_timer;
                WakeupNodeTimer wakeupnode_timer;
                int alpha_num_pkts;
                int beta_num_intervals;
                 friend class PutNodeToSleepTimer;
                friend class WakeupNodeTimer;
#endif // MODIFY_DEREJE
};
```

### A.2.2. Modifications in dsragent.cc

```
#ifdef MODIFY_DEREJE
void
PutNodeToSleepTimer::expire(Event *)
{
 double t2Increment = 0.0;
 a_->node_->PutNodeToSleep();
 if (a_->alpha_num_pkts == 0)
 {
  a_->beta_num_intervals++;
  t2Increment = a_->node_->GetDeltaT2()*a_->beta_num_intervals;
  double maxt2Increment = a_->node_->GetMaxT2Increment();
  if ((t2Increment > 0.0) && (maxt2Increment > 0.0) &&
        (t2Increment > maxt2Increment))
    t2Increment = maxt2Increment;
 }
 else
  a_->beta_num_intervals = 0;
 w_->resched((a_->node_->GetMinSleepTime())+ //t2+beta*delta_t2
            t2Increment);
}
void
WakeupNodeTimer::expire(Event *)
{
 a_->node_->WakeupNode();
```

```
 a_->alpha_num_pkts = 0;
 s_->resched(a_->node_->GetMinWakeTime());
}
#endif // MODIFY_DEREJE

DSRAgent::DSRAgent(): Agent(PT_DSR),request_table(128), \
route_cache(NULL), send_buf_timer(this)
#ifdef MODIFY_DEREJE
, putnodetosleep_timer(this, &this->wakeupnode_timer), wakeupnode_timer(this, &this-
>putnodetosleep_timer), alpha_num_pkts(0), beta_num_intervals(0)
#endif // MODIFY_DEREJE

void
DSRAgent::Terminate()
{
        int c;
        for (c  = 0 ; c < SEND_BUF_SIZE ; c++) {
                if (send_buf[c].p.pkt) {
                        drop(send_buf[c].p.pkt, DROP_END_OF_SIMULATION);
                        send_buf[c].p.pkt = 0;
                }
        }
#ifdef MODIFY_DEREJE
        if (putnodetosleep_timer.status() == TIMER_PENDING) {
                putnodetosleep_timer.cancel();
        }
        if (wakeupnode_timer.status() == TIMER_PENDING) {
                wakeupnode_timer.cancel();
        }
#endif // MODIFY_DEREJE
}

void
DSRAgent::handlePacketReceipt(SRPacket& p)
 /* Handle a packet destined to us */
{
```

```
 hdr_sr *srh = hdr_sr::access(p.pkt);

#ifdef MODIFY_DEREJE
 alpha_num_pkts++;
 double deltat1 = node_->GetDeltaT1();
 double minWakeTime = node_->GetMinWakeTime();
 if (deltat1 == 0)
 {
  putnodetosleep_timer.resched(minWakeTime);
 }
 else
 {
  // Reschedule timer to last_wakeup_time + t1 + (alpha*deltat1) - now
  putnodetosleep_timer.resched(node_->GetLastWakeupTime() +
                               minWakeTime + (alpha_num_pkts*deltat1) -
                               Scheduler::instance().clock());
 }
#endif // MODIFY_DEREJE

 if (srh->route_reply())
  { // we got a route_reply piggybacked on a route_request
   // accept the new source route before we do anything else
   // (we'll send off any packet's we have queued and waiting)
   acceptRouteReply(p);
  }

 if (srh->route_request())
  {
   if (dsragent_reply_only_to_first_rtreq  && ignoreRouteRequestp(p))
         { //we only respond to the first route request
          // we receive from a host
          Packet::free(p.pkt);    // drop silently
          p.pkt = 0;
          return;
         }
   else
```

```
            {
              request_table.insert(p.src, p.src, srh->rtreq_seq());
              returnSrcRouteToRequestor(p);
            }

    }
  if (srh->route_error())
    { // register the dead route
                processBrokenRouteError(p);
    }

  /* give the data in the packet to our higher layer (our port dmuxer, most
   likely) */
  handPktToDmux(p);
}

#ifdef MODIFY_DEREJE
  alpha_num_pkts++;
  double deltat1 = node_->GetDeltaT1();
  double minWakeTime = node_->GetMinWakeTime();
  if (deltat1 == 0)
  {
    putnodetosleep_timer.resched(minWakeTime);
  }
  else
  {
    double resched_time = node_->GetLastWakeupTime() +
                    minWakeTime + (alpha_num_pkts*deltat1) -
                    Scheduler::instance().clock();
    if (resched_time <= 0)
    { // That means the node is sleeping currently and this packet
      // will wake it up, how about an assertion that
      // alpha_num_pkts is 1 ?
      // For now, simply resched the timer to minWakeTime
      putnodetosleep_timer.resched(minWakeTime);
    }
```

```
   else
   {
    // Reschedule timer to last_wakeup_time + t1 + (alpha*deltat1) - now
    putnodetosleep_timer.resched(node_->GetLastWakeupTime() +
                                 minWakeTime + (alpha_num_pkts*deltat1) -
                                 Scheduler::instance().clock());
   }
 }
#endif // MODIFY_DEREJE
```

Modifications made to DSR related files in the distribution of NS are omitted for reducing the size of the report.

# Bibliography

[1] S. Corson and J. Macker, Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, *Request for Comments 2501*, Internet Engineering Task Force, January 1999

[2] Christine E. Jones, Krishna M. Sivalingam, Prathima Agrawal, Jyh-Cheng Chen, A Survey of Energy Efficient Network Protocols for Wireless Networks, *Wireless Networks*, 7(4): 343- 358, July 2001
http://www.cs.pitt.edu/~melhem/courses/3530/papers/pm1.pdf

[3] Laura Marie Feeney, Martin Nilsson, Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment, In *Proceedings of IEEE INFOCOM*, 2001
http://citeseer.ist.psu.edu/feeney01investigating.html

[4] Laura. Marie. Feeney, An Energy Consumption Model for Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks, *Journal of Mobile Networks and Applications*, 6(3): 239-249, June 2001
http://www.ietf.org/proceedings/99jul/slides/manet-feeney-99jul.pdf

[5] D. B. Johnson and D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, In *Mobile Computing*, Chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996
http://www.ics.uci.edu/~atm/adhoc/paper-collection/johnson-dsr.pdf

[6] Josh Broch David A. Maltz David B. Johnson Yih-Chun Hu Jorjeta Jetcheva, A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols, In *Proceedings of Mobile Computing and Networking*, 1998
http://www.ics.uci.edu/~atm/adhoc/paper-collection/johnson-performance-comparison-mobicom98.pdf

[7] Suresh Singh, Mike Woo, CS Raghavendra, Power Aware Routing in Mobile Ad Hoc Networks, In *Proceedings of Mobile Computing and Networking*, 1998
http://apollo.usc.edu/testbed/publications/islped02/P171-maleki.pdf

[8] Sheetalkumar Doshi, Timothy X Brown, Minimum Energy Routing Schemes for a Wireless Ad Hoc Network, to appear in *Proceedings of IEEE INFOCOM*, July 2002
http://ece-ww.colorado.edu/~timxb/timxb/publications/02INFODoshi.pdf

[9] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, Robert Morris: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks, *ACM Wireless Journals*, 8(5), September, 2002
http://citeseer.ist.psu.edu/chen01span.html

[10] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek and M. Degermark, Scenario-based Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks, In *Proceedings of Mobile Computing and Networking*, pp. 195-206, August, 1999
http://www.iponair.de/publications/Yu_ICT2002.pdf

[11] P. J. M. Havinga and G. J. M. Smit, Energy-Efficient Wireless Networking For Multimedia Applications, *Wireless Communications and Mobile Computing*, Wiley, 2001
http://wwwhome.cs.utwente.nl/~havinga/papers/eewn.pdf

[12] T. Brown, Sheetalkumar Doshi, and Q. Zhang. Optimal power aware routing in a wireless ad hoc network, 2001.

[13] Laura Marie Feeney and Martin Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In IEEE INFOCOM, 2001.

[14] M. Subbarao. Dynamic power-conscious routing for manets: An initial approach. IEEE VTC, September 1999.

[15] Ram Ramanathan and Regina Hain. Topology control of multihop wireless networks using transmit power adjustment. In INFOCOM (2), pages 404{413, 2000.

[16] V. Roduplu and T. Meng. Minimum energy mobile wireless networks. In IEEE JSAC, pages 1333{1344, 1999.

[17] Jae-Hwan Chang and Leandros Tassiulas. Energy conserving routing in wireless ad-hoc networks. In INFOCOM (1), pages 22{31, 2000.

[18] Ya Xu, John S. Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad hoc routing. In Mobile Computing and Networking, pages 70{84, 2001.

[19] Qun Li, Javed A. Aslam, and Daniela Rus. Online power-aware routing in wireless ad-hoc networks. In Mobile Computing and Networking, pages 97{107, 2001.

[20] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, and Robert Morris. Span: An energy-e±cient coordination algorithm for topology maintenance in ad hoc wireless networks. In Mobile Computing and Networking, pages 85{96, 2001.

[21] Tamer A. ElBatt, Srikanth V. Krishnamurthy, Dennis Connors, and Son K. Dao. Power management for throughput enhancement in wireless ad-hoc networks. In ICC (3), pages 1506{1513, 2000.

[22] Samir Das, Charles E. Perkins, Elizabeth M. Royer, and Mahesh K. Markina. Performance comparison of two on-demand routing protocols for ad hoc networks. IEEE Personal Communications, 8:16{28, 2001.

[23] S. Narayanaswamy, V. Kawadia, R. Sreenivas, and P. Kumar. Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the compow protocol, 2002.

[24] V. Kawadia, Y. Zhang, and B. Gupta. System services for implementing ad hoc routing protocols, 2002.

# Acronym

| | |
|---|---|
| AODV | Ad hoc On-demand Distance Vector |
| CBR | Constant Bit Rate |
| DSR | Dynamic Source Routing |
| GSM | Global System for Mobile communications |
| ICMP | Internet Control Message Protocol |
| IEEE | Institute of Electrical and Electronics Engineers |
| IETF | Internet Engineering Task Force |
| IP | Internet Protocol |
| LAN | Local Area Network |
| LLC | Logical Link Control |
| MCA | Media Access Control |
| MANET | Mobile Ad hoc NETwork |
| MPR | Multi Point Relay |
| NAM | Network Animator |
| NS | Network Simulator |
| OLSR | Optimized Link State Routing Protocol |
| OSI | Open System Interconnection |
| OTcl | Object Tool Command Language |
| PDA | Personal Digital Assistant |
| RREP | Route REPly |
| RREQ | Route REQuest |
| TCP | Transmission Control Protocol |
| TTL | Time To Live |
| UDP | User Datagram Protocol |