1-1-2003

Simulation study of power aware routing algorithms in multi-hop networks

Kalaimagal Balasubramaniam
Ryerson University

Simulation Study of Power Aware Routing Algorithms in Multi - Hop Networks

By

Kalaimagal Balasubramaniam

A project
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of

Master of Engineering

in the department of
Electrical and Computer Engineering

Toronto, Ontario, Canada, 2003

Author's Declaration:

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Kalaimagal Balasubramaniam

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Kalaimagal Balasubramaniam

Simulation Study of Power Aware Routing Algorithms in Multi - Hop Networks

Abstract

In a multi hop network, the connection between an arbitrary source and destination node can be established using intermediate nodes. Most of these nodes heavily rely on the battery power to keep them-selves up and running. Therefore, the power (or battery lifetime) and the number of intermediate nodes (or delay) play a major role in multi hop networking. In this project we discuss the existing power aware routing algorithms, MHC, MTPR, and MBCR, which are used to minimize the delay, power and nodal over-utilization respectively. This project also presents two newly proposed algorithms, MWBCR and MTP-WBCR and investigates the performance in terms of delay, power usage and residual capacity for homogeneous and heterogeneous nodal capacity scenarios. These power-aware algorithms are simulated in a simulation tool, based on Dijkstra's shortest path algorithm, to evaluate the performance using Java programming language. Simulation study shows that the combined (weighted) MTPR and MBCR algorithm (MTP-WBCR) not only conserves the power and also distributes the routes in the network hence reducing the bottleneck problems at the expense of increased delay. Also it was noted that as the node conservation factor increases, the residual capacities at every node become more balanced.

Borrower List

Ryerson University requires the signatures of all persons using or photocopying this thesis.

Please sign below, and give address and date.

Acknowledgements

I would like to take this opportunity to thank the people who have offered technical insight, and inspiration towards the successful completion of this project.

First of all, I would like to thank Dr. Alagan Anpalagan for believing in me and for all the help that he provided me during the designing of this project. I would like to also thank Mom, my husband Ranjan and my brother, Haran for their special contributions and as well the friends.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Traditional wireless communication was built around cellular architecture where high power base stations were erected to serve an area of interest. It supported mobile users even in areas with no line of sight. On the other hand, there was a need for a set of devices to be connected on demand basis with little or no mobility and in a fairly closer environment, for example, in battlefield communication. This type of network is generally known as multi hop networks. The invention of mobility devices such as cell phone and laptop computers has surged the demand for efficient multi hop networks. In the mobile multi hop networks, data transmission is done by routing protocols through multiple nodes. There are many types of routing protocols as described later in chapter 3. In order to have a better understanding of the multi hop network routing protocols, first of all a novice has to have an understanding of terminologies such as routing and wireless networking.

1.1 Wireless Networking

Wireless networking is the connectivity that is made between two or more computers using network protocols. This connectivity does not contain the network cables that are present in the wired networks. Any type of network that does not contain network cables should be known as wireless network. Currently there are two types of wireless network available in the commercial market. They are network that uses an access point or base station (supported with existing infrastructure) and ad-hoc or peer-to peer networking (set up on demand).

1.1.1. Infrastructure-Based Networking

In this type of network the access point functions like a hub, providing connectivity for the wireless computers. It can connect (or "bridge") the wireless LAN to a wired LAN, allowing wireless computer access to LAN resources, such as file servers or existing Internet connectivity. There are two types of access points. They are hardware access point and software access point.

Wireless devices from different retailers could be mixed and used because all the devices are developed using standards (e.g. IEEE 802.11). Wireless network cannot interact with wired LAN or access shared Internet access. However this discrepancy could be overcome by establishing some kind of bridging between both types of network. The bridging could be accomplished though a hardware or software access point. Hardware access points are only available in one type of network interface. At the same time software access points are available with various types of network interfaces. A software access also has the features such as shared Internet access, web caching or content filtering. These features provide the administrators and users with significant benefits. Typical range of a wireless access point indoor would be 150-300 ft. The outdoor range is around 1000 ft. The range of the wireless access points depends on the environment and speed decreases with distance

Wireless network computers can use multiple access points. Most of the time, separate access points are connected to a wired local area network (LAN) thus provide wireless connectivity to some areas of a building. These access points are in turn connected to a main LAN for access to resources such as file servers.

The advantage of wireless networking is that it facilitates a user to move from one area to the other. This process is known as roaming. When the computer moves from one area to the other, the software and the hardware of the roaming computer maintains the network connection by monitoring the strength of the in-range access points and lock in to the access point that has the best quality. This process is usually transparent to the user. Some access point configurations ask the user to type in the password when he/she moves from one area to the other. It is also important that the wireless areas of various access points should overlap in order to facilitate roaming.

Two local area networks or LAN's could be connected using two access points. This is mainly done in campuses or businesses that are in the immediate proximity but are separated by the public thoroughfare. The installation of this type requires two access points either software access point or hardware access point. These access points act as the bridge between the two LAN's or local area networks thus forming an interconnection between them.

Wireless networks are securely sound but could be eavesdropped by specialty equipment. Wireless cards functions just like Ethernet cards thus their addresses are similar to that of the Ethernet cards.

1.1.2 Ad-Hoc Networking

In this type of network, each of the computers that form the network is equipped with a wireless networking interface card. Every one of those computers can communicate directly with all of the other wireless enabled computers. They can share files and printers this way, but may not be able to access wired LAN resources, unless one of the computers acts as a bridge to the wired LAN using special software (this functionality is known as bridging). Figure 1.1 shows an example of a mobile ad-hoc network

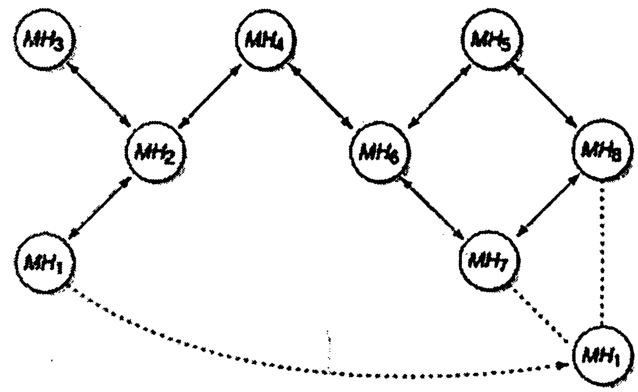


Figure 1.1: An Ad-Hoc Network.

1.2 Routing

Routing is the act of transferring data packets from a source host to the destination host. According to the OSI model, routing takes place at layer 3, which is the network level. The router acts as the intermediate node between the source and the destination host. How routing is done between a source host and a destination host is shown in Figure. 1.2. Routing consists of two main components. They are the (optimal) path determination and transferring information groups (or data packets) through inter-network. The process of transferring data packets through inter-network is also known as switching.

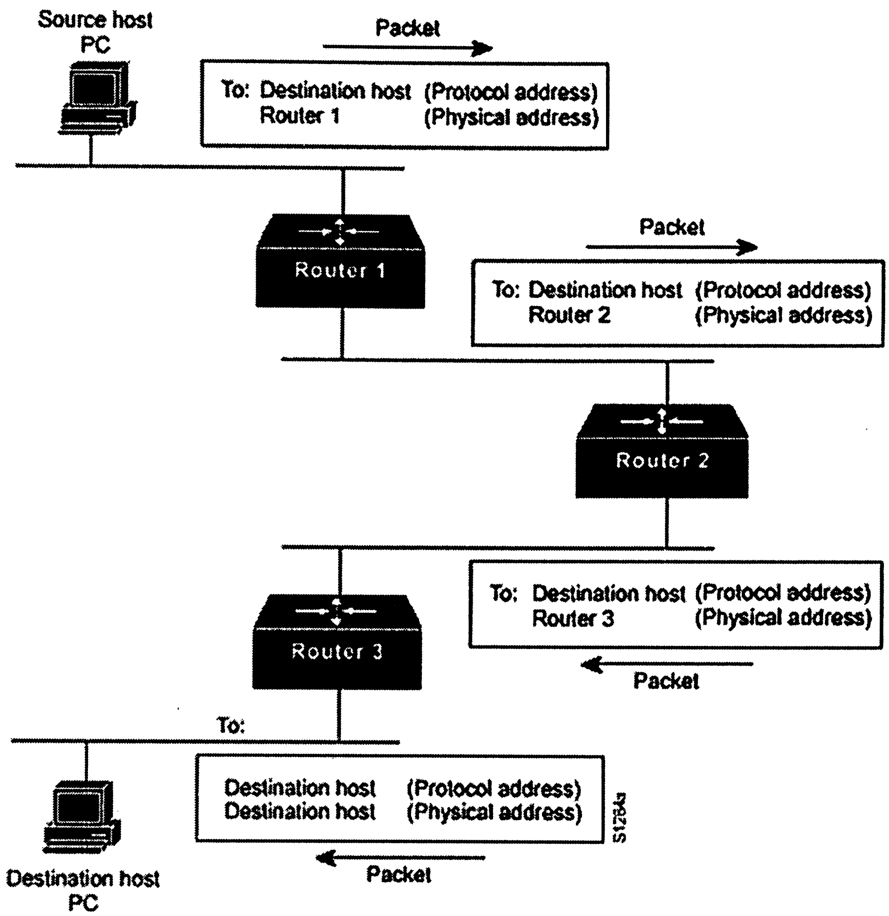


Figure 1.2: Routing between a source host and a destination host [11].

The application layer of a network uses the destination protocol address, to assign the destination node, an address. The router then converts the destination protocol address in to a physical address that is being understood by the physical medium. As it examines the packet's destination protocol address, the router determines that it either knows or does not know how to forward the packet to the next hop. If the router does not know how to forward the packet, it typically drops the packet. If the router knows how to forward the packet, it changes the destination physical address to that of the next hop and transmits the packet.

The next hop may, in fact, be the ultimate destination host. If not, the next hop is usually another router, which executes the same switching decision process. As the packet moves through the Internet work, its physical address changes, but its protocol address remains constant.

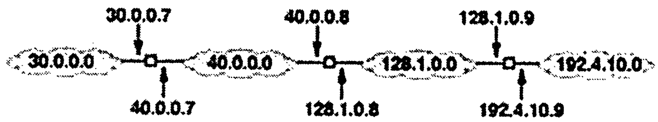
1.2.1 Path Determination

Metric is the measurement unit that is used to measure a path using routing algorithms. In order to get the optimal path the routing algorithms create and maintain a routing table that consists of routing information.

Route information varies depending on the routing algorithm used. Routing algorithms provides the information to the routing tables and thus those tables get filled. Destination/next hop associations notifies a router that a particular destination can be reached optimally by sending the packet to a particular router representing the "next hop" on the way to the final destination. When a router receives an incoming packet, it checks the destination address and attempts to associate this address with a next hop.

Routing tables also can contain other information, such as information about the desirability or likeness of a path. An example of a routing table is shown in Figure 1.3. Routers compare metrics or length of the paths to determine optimal routes, and these metrics differ depending on the make up of the routing algorithm used.

Routers communicate with one another and maintain their routing tables through the transmission of a variety of messages. The routing update message is one such message that generally consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of network topology. A link-state advertisement, another example of a message sent between routers, informs other routers of the state of the sender's links. Link information also can be used to build a complete picture of topology to enable routers to determine optimal routes to network destinations.



Destination	Mask	Next Hop
30.0.0.0	255.0.0.0	40.0.0.7
40.0.0.0	255.0.0.0	deliver direct
128.1.0.0	255.255.0.0	deliver direct
192.4.10.0	255.255.255.0	128.1.0.9

Figure 1.3: An example of a routing table [10].

The routing table is being maintained in the mobile nodes to facilitate in locating route information. In the routing table, the next hop is used to let the router know of the next intermediate node that has the information about the destination node. The masks is used to cover the packet's destination address to prevent the hackers from attacking the packet's travel path and reveal the information that is contained in the packet. The destination is used to let the router know of the destination address (physical address) for a particular packet.

1.2.2 Performance Metrics

The performance metrics are being designed using the basic framework of a mobile multi hop network. The variables such as throughput and delay along with associated mean, variance and distribution are being used as performance metrics for user data. The efficiency of the mobile multi hop network which is calculated based on protocol overhead and data packets delivered correctly is used to measure the integrity of the protocol it self.

Three basic performance metrics are considered in this project for the performance analysis. The **hop count** is used to count the number of hops that are being used to transmit data from a particular source node to the destination node. The **transmitted power** is the term used for the power consumed in transmitting data packets from a source node to the destination node. The **residual capacity** is used to measure the remaining power in a node after the transmission of data packets.

1.3 Motivation

There are two important factors that affect the performance of a mobile multi hop network. They are power budget and power control. The mobile nodes that makeup the multi hop networks operates using battery power. There are two ways the battery power of the mobile nodes gets used. At first the mobile nodes might transfer data to a desired destination node. Second of all a mobile node might want to offer itself as an intermediate forwarding node for data going between a source node and a destination node. The action of node as an intermediate forwarding node is costly in terms of power consumption and also without the availability of such node there will be no multi hop network present.

The decision as to which node should or should not forward traffic may be tricky at first. The solution is that a node with full batter power should act as a forwarding node and a node whose battery power is depleted should only limit its activity to transmitting and receiving only emergency or high priority messages. Server nodes may try to reserve bandwidth to act as source nodes and rely on their neighbors to establish routes with the destination nodes for which the data is intended. Some mobile nodes may use other nodes as forwarding nodes and may not offer anything in return and should be isolated from the network for their bad behavior. As the frequency of periodic data transmissions between nodes increases, the route information cached becomes valid. The validation of the route allows nodes to find the best route without using many control messages to locate a probable route.

There are two aspects that make up the power control schemes in a mobile multi hop network. Reducing the amount of power to the communication interfaces and entering sleep mode are regarded as the two valuable ways in which the battery life of a mobile node can be extended. The power savings techniques are crucial for the functionality of wireless devices such as PDA's and cellular phones.

From the previous works [3][5], we have studied some power aware algorithms that are used to minimize the transmission power and maximize the residual capacity. To the best of our knowledge there are not any algorithms that consider both transmission power and residual capacity as they try to optimize the transmission power being used by the nodes. In this report, we are proposing power aware algorithms, which consider both transmission power and residual capacity. We are also proposing another algorithm in which a weight is being assigned to the residual capacity to measure the performance of the algorithm in a multi hop networks. It is expected that this algorithm will save the over utilized nodes.

The multi-hop communication techniques could also make the communication with a mobile device more difficult in the process. To enable sleep mode, the multi hop routing algorithms require some sort of scheduling algorithm so that the mobile nodes can decide when to exit sleep mode and start listening to the messages. One scheduling technique suggests using synchronization between the wireless nodes. The other scheduling technique calls for the use of reserving a special signaling channel, which can be used as a paging system. The use of the special signaling channel requires allocating bandwidth thus decreases the amount of bandwidth available for data transfers.

1.4 Contribution

In this project we

- study the existing primitive power aware routing algorithms, MHC, MTPR, and MBCR, which are used to minimize the delay, power and nodal over-utilization respectively,
- propose two algorithms, MWBCR and MTP-WBCR that are weighted MBCR and combined (weighted) MBCR and MTP respectively,
- develop a simulation tool using a modified version of Dijkstra's shortest path algorithm to evaluate the performance of power aware algorithms in multi hop networks,
- investigate the performance in terms of delay, power usage and residual capacity for homogeneous and heterogeneous nodal capacity scenarios and for different nodal conservation factors,
- show through simulation studies that the proposed algorithms have potential to distribute the routes in the network and hence reduce the bottleneck problems at the expense of increased delay and power usage.

1.5 Organization

The thesis report has been organized in the following manner. Chapter 1 of the report introduces the purpose of the project as well as set the stage for clear understanding of the rest of the report. Chapter 2 of the report focuses on the multi hop network model as well as the parameters that plays role in the proper functionality of the model. Chapter 3 focuses more on the detailed analysis of the some of the popular power aware algorithms and explains them using numerical examples. Chapter 4 of the report presents the computer simulation of a sample network model written using Java programming language as well as the results produced using the simulation and also discusses the results in a detailed manner. Chapter 5 summarizes the contributions of the project and also focuses on the future directions of studies.

Chapter 2

System Model

In this chapter we consider the multi hop radio network model, its parameters and applications.

2.1 Multi Hop Radio Network Model

In a multi hop radio network, each of the devices that forms the network is equipped with a wireless networking interface card. Every one of those devices can communicate directly with at least one other wireless enabled device. Figure. 2.1, shows a sample model of a multi hop network where source A attempts to communicate to destination E through intermediate nodes B and C. In practice, nodes B and C have to be compensated to participate in the multi hop communication. Also, there is an issue on the security of the communication via other users' devices. We do not consider the above-mentioned issues, rather, concentrate on the routing algorithms and their performance in this work.

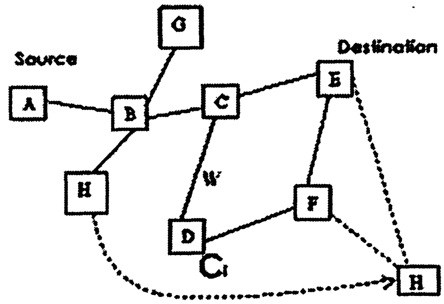


Figure 2.1: Sample Multi hop Network.

Multi hop networks are highly dynamic in nature. The hosts could move within the network or routing domain in any speed or they can disappear and reappear in dynamic fashion. In our study, however we assume that the hosts are stationary. The interconnection between multi hop hosts is established using the multi hop routing protocol. The multi hop wireless networks are susceptible to multi-path fading, noise, and signal interference. Hence, the routing or path selection in wireless multi hop networks is a challenging problem. For simplicity, we assume distance dependent path loss model between two communication nodes.

Multi hop networks are capable of providing multi-hop communications. The multi-hop functionality allows the mobile nodes that cannot reach the destination nodes to rely on other nodes that are near by, to transfer the information through. The mobile nodes are highly reliant on batteries for proper operation and thus the batteries on the wireless nodes play an important role on the overall performance of the network.

Some of the notations used in this report are described below.

S: source Node
D: sestination Node
W: weights
W_t: sum of the weights
d : the distance between the nodes
ψ_j : pre-detection threshold at node n_j
P_j : transmission power of node n_j
$G_{i,j}$: path gain between hosts n_i and n_j
η_j: thermal noise at host n_j
C_i: battery capacity of node n_i
l : source sestination pair or SD pair
R_l : route for l
N_{R_l} : number of hops in R_l
A_{R_l} : set of all possible routes for R_l
$Cost_{R_l}$: Cost for route R_l where cost $\in$ {hop count, transmitted power , residual capacity}

The link cost in a mobile multi hop network is also known as link weight. Weight (W) is assigned to each radio connection between two nodes. The value of W reflects the quality of service of that link. In this work, the link weight W is the distance dependant path loss between any two nodes, and/or capacity at any specific node depending on the algorithm under consideration. Other properties can also be mapped to W such as: bit error rate, number of hops or collisions rate.

Information is stored in every node about W for each outgoing link. The routing algorithm attempts to find a path through the network; it calculates the sum of the weights (W_t) for different paths from source to destination. The path with the best W_t is chosen as a route to achieve the required performance.

The network model can be described as follows. The network consists of N nodes. The source node and destination node are present in the network and are randomly selected. The packets are transferred from the source node using intermediate nodes that were created using path selection algorithms and that satisfied over performance metrics (i.e., minimize delays).

Radio propagation is modeled in our work using the $1/d^{\alpha}$ attenuation model with roll off factor α (α is usually 2 for short distance and 4 for long distance) where d is the distance between the nodes. The signal to noise ratio (SNR) received at a node n_j should be greater than a pre-detection threshold of ψ_j in order to have a successful communication. For successful transmissions from a host n_i to n_j (i.e., reverse link communication), the SNR at host n_j should satisfy (1) which defines the relationship between ψ_j and SNR at host n_j as:

$$\mathrm{SNR}_j = \frac{P_i G_{i,j}}{\sum_{k \neq i} P_k G_{k,j} + \eta_j} \geq \psi_j \quad (1)$$

where P_i is the transmit power, $G_{i,j}$ is the path gain between hosts n_i and n_j. Thermal power is assumed to be negligible since it is small compared to the interference power (as usually the case in CDMA systems which are interference limited rather than noise limited).

Power control plays a major role in reverse link communication of any CMDA based systems. It controls the amount of transmission power just enough to maintain the required SNR at the receiver. In the CDMA network with the homogeneous users (i.e., equal required data rate and SNR), power control sets the required power to be equal for all users. Following the same approach, we assume that the received power at each (intermediate or end) node to be equal in the multi hop network. For simplicity it is assumed to be 1 in our work.

In this project we consider a distance dependant path loss model, where we assume that the received power, $P_r = [P_t (K)]/ d^{\alpha}$. In the above-mentioned relationship, P_t is known as the transmitted power and could be equated as follows, $P_t = [P_r (d^{\alpha})] / K$. In a power controlled multi hop network we assume that P_r to be 1. For simplicity it is assumed that K to be 1 which results in the following equation, $P_t = d^{\alpha}$, α=4. The farther the source node is from the destination node, which results in the more transmission power. The residual capacity is also being used as a link weight and is described using the relationship $1/C_i^{\beta}$ where $\beta = 1,2,3$.

The paths are created using the next hop node that is present in the routing table of an intermediate node. The delay is the time taken for the data packet to be transferred from the source node to the destination node. The transmission power is the amount of power used for the data transfer purposes between participating nodes. The residual capacity is the power that is left on a node after the data transfer. The network model that is being used in this project is known as tilted mesh.

We consider two types of nodal capacity models, heterogeneous and homogenous to get a clear picture of the performance of the algorithms. In heterogeneous model, the residual capacity is being assigned as a uniform distribution of random values in the nodes. In homogeneous model, the residual capacity is constant among nodes. For the nodal models, the simulation is done for the packet transmission at 1, 5 and 10 packets and being regarded as levels in some parts of the report.

2.2 Application: Sensor Networks

One of the emerging applications using multi hop communication technology is the sensor network. A sensor network is made up of high number of sensors, which are stretched across a geographical area. Every sensor in the network is capable of communicating with each other through wireless means and sufficient intelligence for signal processing of the data. The examples below list some of the common sensor networks that have been currently in use.

1. Wireless traffic sensor networks that are used to monitor traffic congestion as well as vehicular movement in a highway or local roads.
2. Wireless surveillance sensor networks, which are used to provide security in shopping malls, factories or other facilities.
3. Wireless parking lot sensor networks are used to find out whether which parking spots are being occupied and which ones are free.

4. Military sensor networks are mainly used to detect the movements of the enemy, which includes movement of enemy convoys as well as foot soldiers, and also to detect the presence of weapons of mass destruction such as poison gases and nuclear weapons.
5. The environmental sensor networks are commonly deployed in plains of deserts, mountains and ocean surfaces to detect and monitor environmental changes.

There are two ways available to classify sensor networks, which are as follows:

1. Whether or not the node individually addressable.
2. Whether the data in network is aggregated.

The sensor nodes in a parking lot network should be individually addressable so that once can determine the locations of the free spaces. It may be necessary to broadcast a message to all the nodes in this particular application. The address ability may not be important if the application requires for the monitoring of temperature in a particular room. The ability of the sensor network to aggregate the data collected in turn could reduce the number of messages that need to be transmitted across the network.

The most important task of a sensor network is to disseminate the data that has been collected to the proper end users. Sensor requirements include the following:

- Greater number of sensors (mostly stationary) for accurate communication
- Low energy use for cost effectiveness and reduced interference
- Network self organization to handle mobility and link disturbance

The performance metrics commonly considered in the multi hop networking are delay and transmitted power. In the next chapter we discuss algorithms that minimize these metrics.

Chapter 3

Power Aware Routing Algorithms

There are many different power aware algorithms proposed in the literature [3][5]. This section contains the discussions of the existing power aware algorithms and two newly proposed algorithms as well. We propose the Minimum Weighted Battery Cost Routing (MWBCR) and the Minimum Transmitted Power and the Weighted Battery Cost Routing (MTP-MWBCR) techniques.

3.1 Existing Power Aware Routing Algorithms

This section of the report analyzes the existing power aware algorithms in detail. The algorithms that are being analyzed are Minimum Hop Count Routing or MHC, Minimum Total Transmission Power Routing or MTPR, and Minimum Battery Cost Routing or MBCR.

3.1.1 Minimum Hop Count Routing (MHCR)

The hop count is the basic performance metric in multi hop networks. This is not a power aware algorithms since it gives no consideration on to the power used by the nodes or power stored in the node. However, it is important since it indicates the delay incurred in any communication. For real time communication (i.e. battlefield), this metric is very important.

At first the algorithm determines the path, which has the minimum hop count. It also determines the route that flows over with the above-mentioned path. The minimum hop routing algorithm uses the Dijkstra's shortest path algorithm [3]. If there is more than one route from the ource to the destination then the route, which has the minimum number of hops, is selected. That is

$$H_{R_l} = \sum_{i \in R_l} i \qquad (2)$$

$$MHCR_l = \min_{R_l} \left\{ H_{R_l} \right\} \qquad (3)$$

where R_l is the route for l and l is the source-destination pair. N_{Rl} is number of hops in R_l. The hop count is directly related to delay. To measure the performance of the packet transmission the basic metric, which is delay, is not enough and thus we also need to consider the other performance metric that is transmission power used in communication.

3.1.2 Minimum Total Transmission Power Routing (MTPR)

The transmission power is generally dependent on interference, noise, distance between hosts and sustainable bit rate. Transmission power can be increased to combat interference and noise and hence reduce the bit errors. However, it will cause added interference to other users in wireless communication through the propagation in free space. So we need to keep the transmission power under control.

The route with the total minimum power could be derived using the transmission power P (n_i,n_j) between hosts n_i and n_j. The total transmission power for an SD pair l could be calculated using (3)

$$P_{R_l} = \sum_{i \in R_l} P_i \quad (4)$$

$$MTPR_l = \min_{R_l}\left\{P_{R_l}\right\} \quad (5)$$

Equation (3) helps obtain a desired route k where A is the set containing all possible routes. The above equation could be solved using shortest path algorithms such as Dijkstra or Bellman-Ford[3]. In our simulation, the Dijkstra's shortest path algorithm is modified to obtain the desired path. As noted in [3] with the number of nodes increases, the end-to-end delay also increases. Since the Dijkstra's algorithm selects routes with more hops, from the standpoint of minimum number of hosts, it is not attractive. To overcome this problem, the transmission power was considered as a cost metric, and the distributed Bellman-Ford algorithm was used [3], but in our case we used the Dijktra's shortest path algorithm.

We have considered the performance metrics, delay and transmission power and they are not simply adequate to measure the performance of the packet transmission. Also, we have to take in to consideration of another performance metric, which is residual capacity.

3.1.3 Minimum Battery Cost Routing (MBCR)

The total transmission power is an important metric since it deals with the lifetime of mobile hosts. This metric reduces the total power consumption of the overall network and it does not reflect on the lifetime of each host. If the minimum total transmission power routes are via a specific host, the battery of this host will replenish quickly. It is a great idea to describe the life time of each host using the remaining battery capacity.

Let C_i be the battery capacity of a host n_i at an instant. The battery cost function of a host n_j is defined as $f_i(c_i)$. Now, suppose a node's willingness to forward packets is a function of its remaining battery capacity. The less capacity it has, the more reluctant it is to forward. One possible choice for f_i is

$$f_i(c_i) = 1 / c_i$$

As the battery capacity decreases, the value of cost function for node n_i will increase. The battery cost R_j for route 1, consisting of D_j nodes, is

$$C_{R_l} = \sum_{i \in R_l} f_i(C_i) \qquad (6)$$

where A is the set containing all possible routes.

Therefore, to find a route with the maximum remaining battery capacity, the route R_1 which has the minimum battery cost should get selected.

$$MBCR_l = \min_{R_l} \{ C_{R_l} \} \qquad (7)$$

The battery capacity is directly incorporated in to the routing protocol and this metric prevents the hosts from being used over a certain limit. Therefore the life time of the host and the time until the network is partitioned are greatly increased. Suppose that all nodes have the same amount of battery capacity, this metric will select the shorter hop route. Since only the summation of values of battery cost functions is being considered, a route which has little battery capacity hosts will still be selected as noted in [3]. The proposed algorithms are being put forward to compare the results with the existing algorithms and also to analyze whether the performance of packet transmission improves.

3.2 Proposed Power Aware Algorithms

This section of the report analyzes the power aware algorithms that have been proposed by the author of this report. The algorithms being analyzed are Minimum Weighted Battery Cost Routing or MWBCR and Minimum Transmission Power and Weighted Battery Cost Routing or MTP-WBCR.

MHCR does not consider the to be used battery power or available battery power. MTPR does consider the to be used battery power but neglects the available battery power at nodes. MBCR considers the available battery power but neglect the to be used battery power. Hence we propose to consider both to be used and currently available battery power by combining both MTPR and MBCR. Also we noted in our preliminary investigation, that MBCR with $1/C_i$ cost function does not save the over utilize nodes from being used further. Hence we propose to give weight to the cost function by considering the cost function as $1 / C_i^\beta$ where $\beta = 1,2,3$. Note that $\beta = 1$ corresponds to MBCR.

3.2.1 Minimum Weighted Battery Cost Routing (MWBCR)

Minimum weighted battery cost routing is based on MBCR. In the minimum weighted battery cost routing (MWBCR) algorithm, the cost is calculated using the formula that is given below:

$$WC_{R_l} = \sum_{i \in R_l} \left\{ \frac{1}{C_i^\beta} \right\} \text{, where } \beta = 3 \text{ (in this case)} \qquad (8)$$

$$MWBCR_l = \min_{R_l} \left\{ WC_{R_l} \right\} \qquad (9)$$

In MBCR algorithm , if one of the mobile nodes were to become a bottleneck (because it is used in most of the routes), that particular node's capacity will get drained soon. As a mean to reduce the bottleneck problem encountered above and to provide a capacity balance for all the nodes, MWBCR employs a variable known as β. The cost equation of a node *i* using the MWBCR becomes as follows:

$Cost_{R_i} = 1/C_i^{\beta}$, Where β is referred to as conservation factor and β = 1,2,3... in our case.

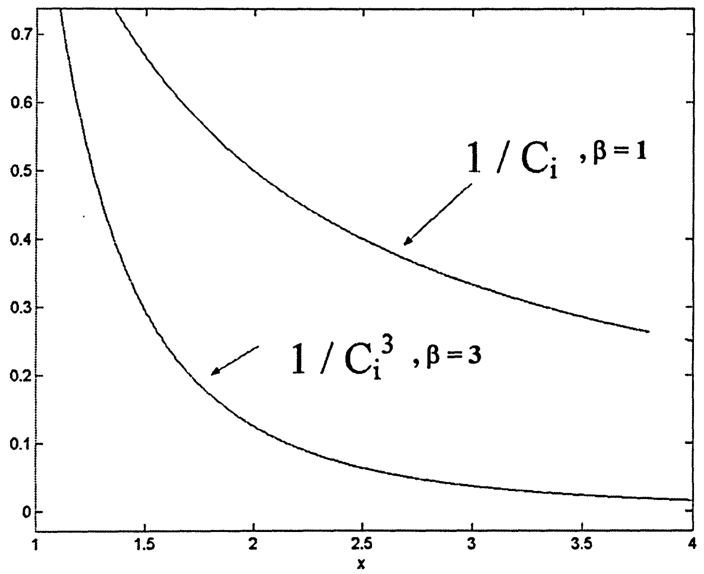


Figure: 3.1 Behavior of the conservation factor β.

Due to the inverse relationship between the capacity and the cost, as the capacity increases, the cost decreases in a non-linear manner. When β = 3 (MWBCR), the cost is smaller than that of β = 1. As the beta increases, the cost drops in a more rapid manner. This behavior leads to the selection of a higher capacity node for data transfer in the case of MWBCR. We expect that more nodes will be used in the routing giving rise to more balanced use of nodes in the network.

3.2.2 Minimum Transmission Power and Weighted Battery Cost Routing (MTP-WBCR)

The minimum transmission power and weighted battery cost routing combines both the minimum battery cost routing and the minimum total transmission power routing. The minimum battery cost routing helps minimize the amount of battery power that is being used when a mobile host is being used as a pathway between a source and a destination node. The capacity or C_i plays a significant role in the reduction of battery power being used and as the capacity increases, the used battery power decreases in multiple folds. The minimum total transmission power routing helps reduce the transmission power. As the transmission power and the battery cost decreases, it could be said that the life of the battery would increase by a gradual amount. Hence, the cost function becomes

$$PC_{R_l} = \sum_{i \in R_l} \frac{d_i^{\alpha}}{C_i^{\beta}} \quad , \text{ where } \alpha = 4, \beta=3 \qquad (10)$$

$$MTPWBCR_l = \min_{R_l}\left\{PC_{R_l}\right\} \qquad (11)$$

In this case however, the numerator in the cost function becomes dominant thus leads to a behavior similar to that of MTPR cost function.

3.3 Numerical Examples

This section explains the previously discussed power-aware algorithms using numerical examples, which are based on a single network topology and is given in Figure. 3.2. There are 6 nodes in the network. Source and Destination are denoted as S and D. There are 3 routes between S and D.

Capacity (C_i) (Residual Capacity) at nodes i : The distances between the nodes are as follows:

S: 0.5	W: 0.5
X: 0.25	Z: 0.2
Y: 0.5	D: 0.5

SX: 1	SZ: √3
XY: √3/2	ZD: √3/2
YD: 1	SW: 1

There are three routes from S to D, namely, R1, R2, R3.

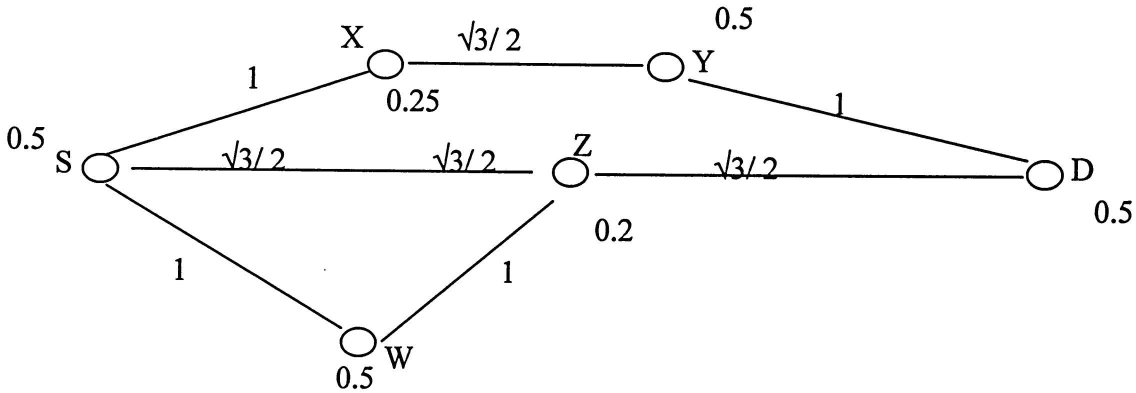


Figure 3.2: Sample network for the numerical example.

Alg. 1: Minimum Hop Count Routing

Three routes from Source node S to the Destination node D are :

R1: S – W – Z – D
R2: S – Z – D
R3: S – X – Y – D

In this case, R2 has minimum number of nodes involved in the path therefore the route R2 will be selected.

Alg. 2: Minimum Transit Power Routing

The received power $P_{r,i}$ at every node i in the network is $P_{r,i} = P_{t,i} * G_i$ where $P_{t,i}$ is the transmitted power at the previous-hop node i and G_i is the link gain for the node i. Due to the expected line of sight communication in multi hop networks, it is assumed

$$G_i = 1 / d_i^4.$$

Now we can write the received power as:

$$P_{r,i} = 1 / d_i^4 * P_{t,i}$$

$$P_{t,i} = P_{r,i} * d_i^4$$

If we simplify the above equation it becomes $P_t = Pd^\alpha$ where P is the minimum power that has to be received by every node to maintain a minimum SNR at the receiving node. Hence, assuming $P_{r,i} = P = 1$, $Cost_i = 1/ d_i^4$. Therefore the costs in different routes are shown below.

R1: $(1^4 + 1^4 (\sqrt{3}/2)^4) = 2.5$
R2: $((\sqrt{3})^4 + (\sqrt{3}/2)^4) = 9.5$
R3: $(1^4 + (\sqrt{3}/2)^4 + 1^4) = 2.5$

In this situation either the route R1 or route R3 will be selected.

Alg. 3: Minimum Battery Cost Routing

The cost function of the MBCR or minimum battery cost routing is given by the equation as follows: $Cost_i = 1 / C_i$

The table below lists all possible routes and their respective costs.

R1: (1/0.5 + 1/0.5 + 1/0.2) = 9
R2: (1/0.5 + 1/0.2) = 7
R3: (1/0.5 + 1/0.25 + 1/0.5) = 8

In this situation the route R2 will be selected since it has the lowest cost.

Alg. 4: Minimum Weighted Battery Cost Routing

The equation below describes the cost function for this algorithm.

$Cost_i = 1 / C_i^3, \beta = 3$

The table given below displays, the routes and their corresponding cost for the MWBCR algorithm.

R1: $(1/0.5^3 + 1/0.5^3 + 1/0.2^3) = 141$
R2: $(1/0.5^3 + 1/0.2^3) = 133$
R3: $(1/0.5^3 + 1/0.25^3 + 1/0.5^3) = 80$

The route R3 will be selected since it has the lowest cost.

Alg. 5: Minimum Transmitted Power and Weighted Battery Cost Routing

The cost function for the algorithm is given by the equation below:

$Cost_i = d_i^4 / C_i^3$

R1: $(1^4/0.5^3 + 1^4/0.5^3 + (\sqrt{3}/2)^4/0.5^3) = 20.5$
R2: $((\sqrt{3})^4/0.5^3 + (\sqrt{3}/2)^4/0.2^3) = 142.3$
R3: $(1^4/0.5^3 + (\sqrt{3}/2)^4/0.25^3 + 1^4/0.5^3) = 52$

From the table above it could be seen that R1 has the minimum cost for the MTPWBCR algorithm and there fore R1 will be selected in this situation.

From the examples we could observe that different algorithms select different routes depending on the cost function used in the algorithms. The summary of all routes selected for each algorithm is given in the table below.

Algorithms	Route
MHCR	R2: S – Z – D
MTPR	R1: S – W – Z – D Or R3: S – X – Y – D
MBCR	R2: S – Z – D
MWBCR	R3: S – X – Y – D
MTPWBCR	R1: S – W – Z – D

Table 3.1: Summary of the routes selected in all the algorithms.

Next, we simulate all the five algorithms discussed in the section to evaluate the performance in terms of delay, transmitted power and residual capacity.

Chapter 4

Performance Evaluation through Simulation

To verify the performance a computer simulation* was done using Java. The Simulation setup and the results are discussed in this chapter.

4.1 Simulation Setup

Simulations were performed using JAVA programming language. All of the above mentioned routing algorithms were implemented using the Dijkstra's shortest path algorithm with appropriate cost metrics. The network that was used in the simulation algorithm consisted of 36 mobile nodes. The network was configured using the tilted mesh topology as seen on figure 4. The simulation was performed for 100 randomly selected source-destination pairs. There were two models taken in to consideration for the simulation of the algorithms. They are heterogeneous distribution of capacity and homogeneous distribution of capacity. For both models, the distances between the nodes were uniformly distributed random double value between 0 and 1. For the heterogeneous model, the capacity was a uniformly distributed integer between 10 and 20. For the homogeneous model, the capacity was a constant value of 15. The integer between 10 and 20 are chosen in the heterogeneous model so that there will be enough capacity to support transmission of 10 packets.

---

* The source code is taken from [8][9][10] and modified to suit the author's needs.

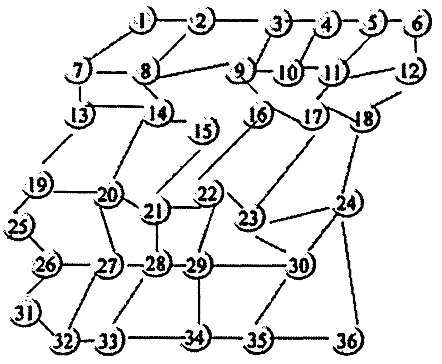


Figure 4: The tilted mesh topology.

At each run the delay at every node (number of hops), transmitted power and residual capacity (remaining capacity at each node) are calculated for all the algorithms. An assumption was made that the delay at every node is 1 unit. Therefore the number of hops is equivalent to the delay. To get clear understanding of the average values of above mentioned performance metrics, the numbers of packets were increased. The 1, 5 and 10 packet transmission scenarios were considered for the both homogeneous and heterogeneous models. The capacity was deducted by the appropriate transmitted power from the associated node after every packet transmission for all the algorithms.

4.2 Simulation Process

- The link weights are being read from the input prompt and being substituted in to the graph object in the simulation program along with the appropriate link.
- The link weights should be appropriately entered for every algorithm along with their proper cost in to the Dijkstra's algorithm so that the shortest path could be calculated.
- For every algorithm's resultant shortest path, the delay, transmission power and residual capacity are calculated after every packet transmission and are stored in a global variable.

- The performance metrics are calculated for all algorithms and are referred to as average values and are graphed to compare the results.
- The graphs and the comparison of graphed results are described in detail under the simulation results section of this report.

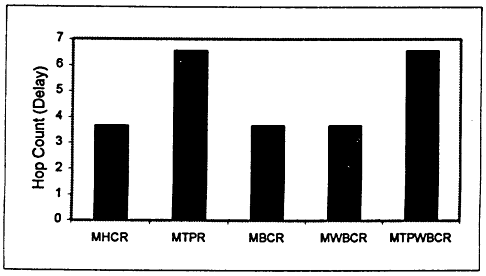
4.3 Simulation Results and Discussion

The simulation results are described in the following sections for the average delay, average transmitted power and residual capacity for both the heterogeneous model and homogenous model.

4.3.1 Delay

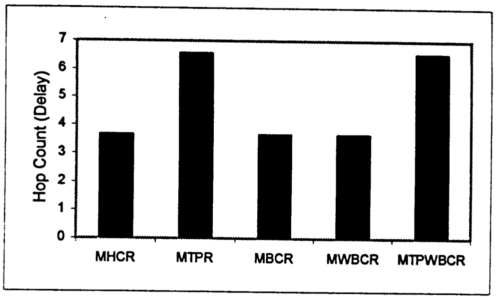
Homogenous Capacity Model

The graphs in Figure 4.1 analyze the delays of various algorithms. In Figure 4.1 a, the MHCR, MBCR and MWBCR algorithms have the lowest delay compared to MTPWBCR, which has the highest value. In Figure 4.1 b, the MHCR algorithm has the lowest delay compared to MTPWBCR, which has the highest value. In Figure 4.1 c the MHCR algorithm has the lowest delay compared to MTPWBCR, which has the highest value. The observations are in parallel with what was expected of the behavior of the algorithms.



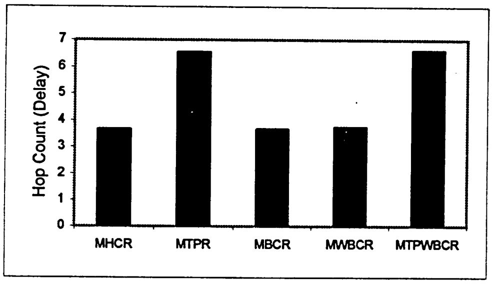
Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.65
MWBCR	3.65
MTPWBCR	6.55

Figure 4.1 a) Average Delay - Homo - Level 1.



Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.65
MWBCR	3.66
MTPWBCR	6.53

Figure 4.1 b) Average Delay - Homo - Level 5.



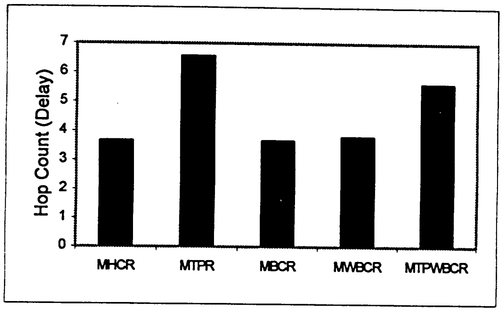
Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.66
MWBCR	3.73
MTPWBCR	6.60

Figure 4.1 c) Average Delay - Homo - Level 10.

<u>Heterogeneous Capacity Model</u>

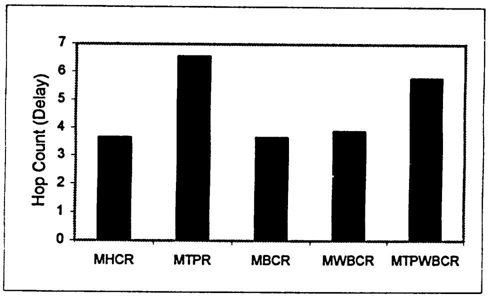
The graphs in Figure 4.2 describe the average delay for each algorithm that has been tested. In Figure 4.2 a), the MHCR algorithm has the lowest delay of all analyzed algorithms which was expected of by theoretical means. The MTPWBCR algorithm has the highest value of all algorithms. This result was also expected by theoretical means. Algorithms MTPR and MWBCR have values for delay which is more than what was expected using the theoretical means. In Figure 4.2 b) MHCR algorithm has the lowest delay compared to MTPWBCR, which has the highest value. In Figure 4.2 c) MHCR algorithm has the lowest delay compared to

MTPWBCR, which has the highest value. These observations are in terms with what was expected of the behavior of the algorithms.



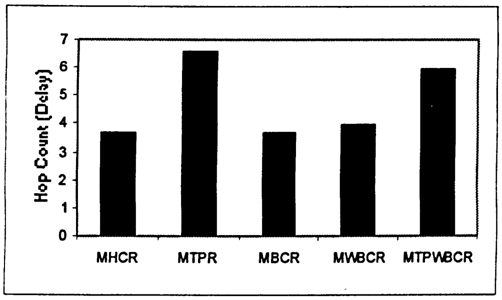
Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.65
MWBCR	3.79
MTPWBCR	5.61

Figure 4.2 a) Average Delay – Hetro – Level 1.



Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.65
MWBCR	3.87
MTPWBCR	5.78

Figure 4.2 b) Average Delay - Hetro - Level 5.



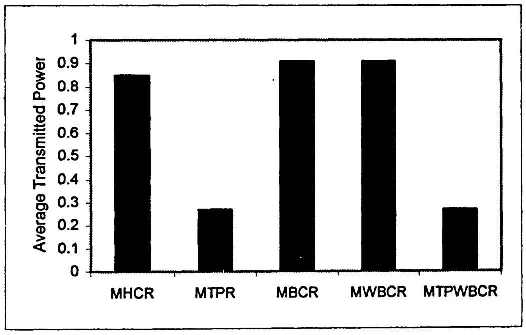
Algorithms	Average
MHCR	3.65
MTPR	6.55
MBCR	3.67
MWBCR	3.94
MTPWBCR	5.96

Figure 4.2 c) Average Delay - Hetro - Level 10.

4.3.2 Transmitted Power

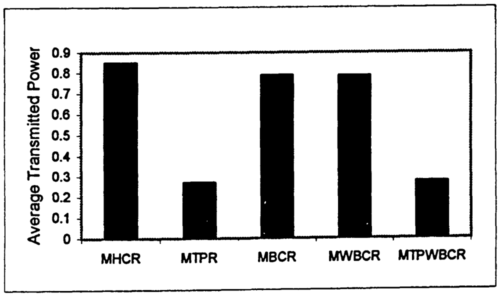
Homogenous Capacity Model

The graphs in 4.3 display the average transmitted power for analyzed algorithms. In Figure 4.3 a, the MHCR algorithm has the highest value compared to MTPR and MTPWBCR algorithms, which have the lowest value. In Figure 4.3 b, the MWBCR algorithm has the highest value compared to MTPR algorithm, which has the lowest value. In Figure 4.3c the MWBCR algorithm has the highest value compared to MTPR algorithm, which has the lowest value. The results displayed in the graphs were similar to what was expected.



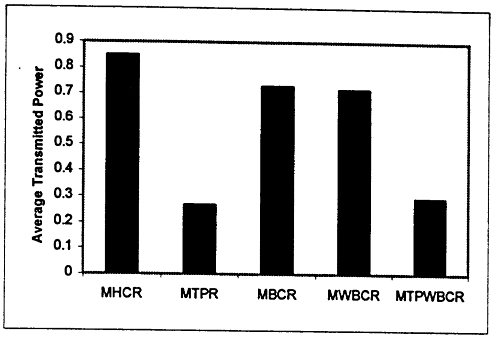
Algorithms	Average
MHCR	0.8479
MTPR	0.2682
MBCR	0.9079
MWBCR	0.9079
MTPWBCR	0.2682

Figure 4.3 a) Average Transmitted Power - Homo - Level 1.



Algorithms	Average
MHCR	0.8479
MTPR	0.2683
MBCR	0.7866
MWBCR	0.7847
MTPWBCR	0.2739

Figure 4.3 b) Average Transmitted Power - Homo - Level 5.

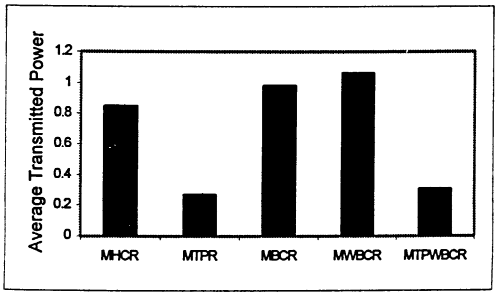


Algorithms	Average
MHCR	0.8479
MTPR	0.2683
MBCR	0.7292
MWBCR	0.7172
MTPWBCR	0.2938

Figure 4.3 c) Average Transmitted Power - Homo - Level 10.

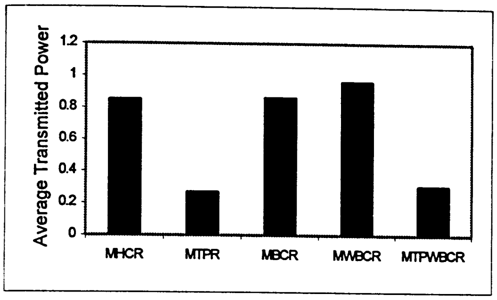
Heterogeneous Capacity Model

The graphs in 4.4 show the average transmitted power for algorithms that have been tested. In Figure 4.4 a, the MHCR algorithm has the highest value compared to MTPR algorithm, which has the lowest value. In Figure 4.4 b, the MWBCR algorithm has the highest value compared to MTPR algorithm, which has the lowest value. In Figure 4.4 c, the MWBCR algorithm has the highest value compared to MTPR algorithm, which has the lowest value. The graphs displayed results, which were expected of the algorithms.



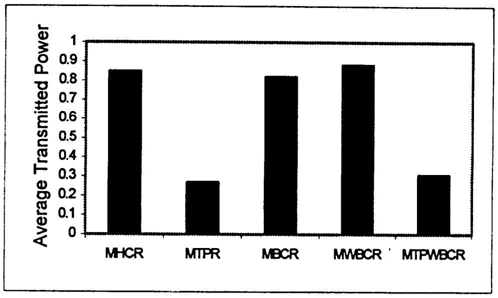
Algorithms	Average
MHCR	0.8479
MTPR	0.2682
MBCR	0.9811
MWBCR	1.0611
MTPWBCR	0.3045

Figure 4.4 a) Average Transmitted Power - Hetro - Level 1.



Algorithms	Average
MHCR	0.8479
MTPR	0.2683
MBCR	0.8587
MWBCR	0.9589
MTPWBCR	0.3036

Figure 4.4 b) Average Transmitted Power - Hetro - Level 5.



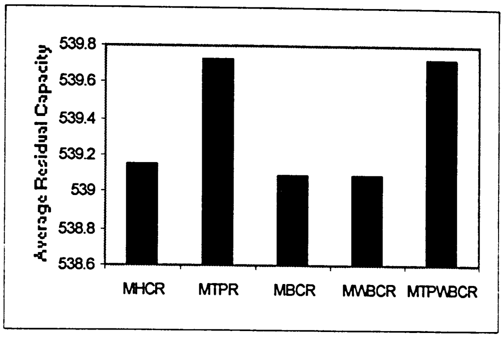
Algorithms	Average
MHCR	0.8479
MTPR	0.2683
MBCR	0.8183
MWBCR	0.8804
MTPWBCR	0.3057

Figure 4.4 c) Average Transmitted Power - Hetro - Level 10.

4.3.3 Residual Capacity

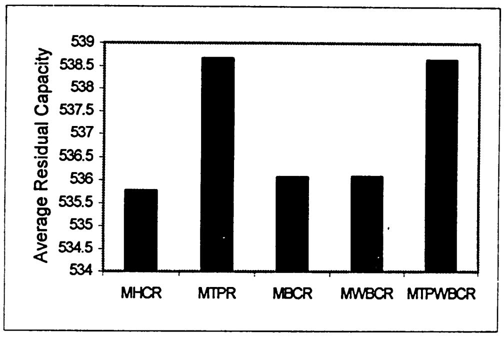
Homogenous Capacity Model

The graphs in Figure 4.5 displays the average residual capacity for the algorithms that were being analyzed in the paper. In Figure 4.5 a, the MTPWBCR and MTPR algorithms have the highest value at the same time, the MHCR, MBCR and MWBCR algorithms have the lowest value of all of the simulated algorithms. In Figure 4.5 b, the MTPWBCR algorithm has the highest value at the same time the MHCR algorithm has the lowest value of all of the simulated algorithms. The same is true in for Figure 4.5 c. The results in the graphs are similar to what was expected.



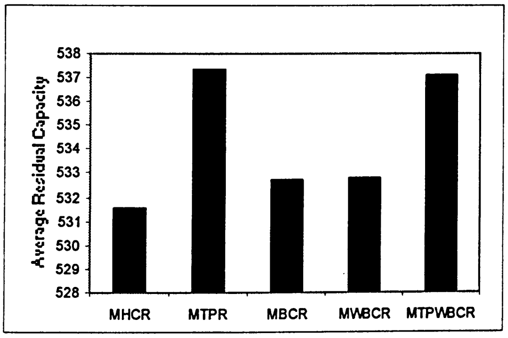
Algorithms	Average
MHCR	539.152
MTPR	539.733
MBCR	539.092
MWBCR	539.092
MTPWBCR	539.732

Figure 4.5 a) Average Residual Capacity - Homo - Level 1.



Algorithms	Average
MHCR	535.7604
MTPR	538.6585
MBCR	536.0668
MWBCR	536.0763
MTPWBCR	538.6306

Figure 4.5 b) Average Residual Capacity - Homo - Level 5.

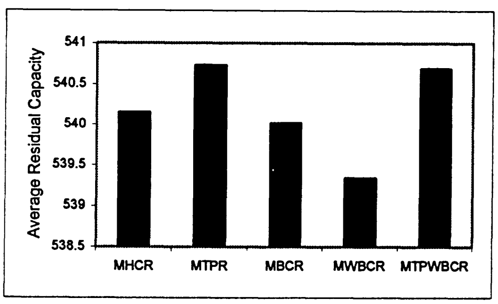


Algorithms	Average
MHCR	531.5209
MTPR	537.3171
MBCR	532.7078
MWBCR	532.8277
MTPWBCR	537.0617

Figure 4.5 c) Average Residual Capacity - Homo - Level 10.

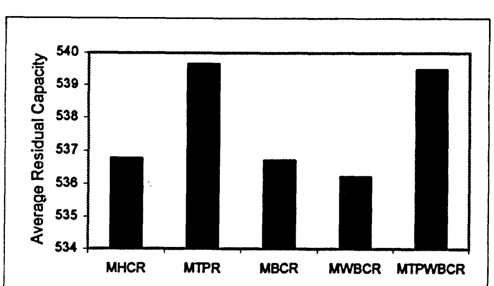
Heterogeneous Capacity Model

This graphs Figure 4.6 describes the average residual capacity for the simulated algorithms. In Figure 4.6 a, the MTPWBCR algorithm has the highest value at the same time the MHCR algorithm has the lowest value of all of the simulated algorithms. In Figure 4.6 b, the MTPWBCR algorithm has the highest value at the same time the MHCR algorithm has the lowest value of all of the simulated algorithms. In Figure 4.6 c the MTPWBCR algorithm has the highest value at the same time the MHCR algorithm has the lowest value of all of the simulated algorithms. The graphs contained information which were as expected in the theoretical analysis of the algorithms.



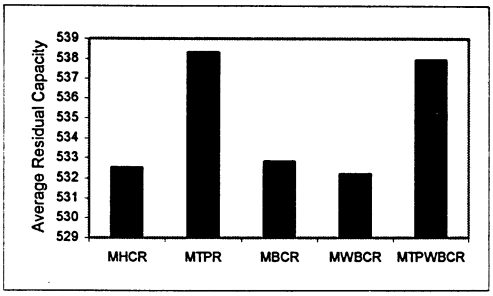
Algorithms	Average
MHCR	540.152
MTPR	540.732
MBCR	540.019
MWBCR	539.339
MTPWBCR	540.695

Figure 4.6 a) Average Residual Capacity - Hetro - Level 1.



Algorithms	Average
MHCR	536.7604
MTPR	539.6585
MBCR	536.7066
MWBCR	536.2053
MTPWBCR	539.4818

Figure 4.6 b) Average Residual Capacity - Hetro - Level 5.

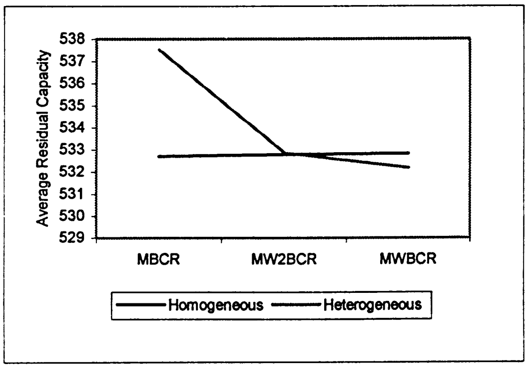


Algorithms	Average
MHCR	532.5209
MTPR	538.3171
MBCR	532.8169
MWBCR	532.1964
MTPWBCR	537.9427

Figure 4.6 c) Average Residual Capacity - Hetro - Level 10.

4.3.4 Average Residual Capacity with Different Weights (Nodal Conservation Factors)

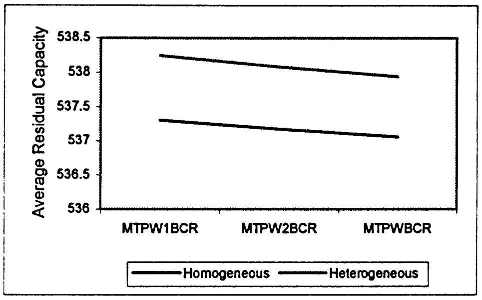
In this section we take a look at average residual capacity with different weights for the MBCR algorithm to see the performance of heterogeneous and homogeneous models. The Figure 4.7 displays the behavior of homogeneous and heterogeneous capacity models for different beta values. The behavior of the heterogeneous capacity model is not constant, and on the other hand, the homogenous capacity model has a positive slope and linear relationship.



Algorithms	Homo	Hetro
MBCR ($\beta = 1$)	532.7078	537.5376
MWBCR ($\beta = 2$)	532.7848	532.8509
MWBCR ($\beta = 3$)	532.8277	532.1964

Figure 4.7 Average Residual Capacity – Homo level 10 ($\beta = 1, 2, 3$).

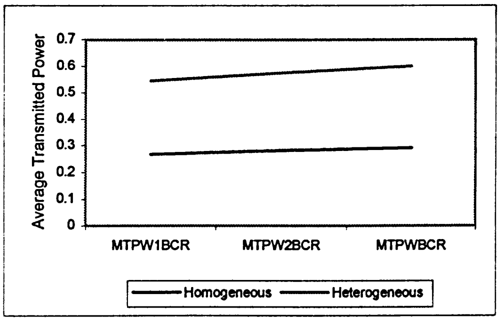
The Figure 4.8 displays the behavior of homogeneous and heterogeneous capacity models for different beta values. The behavior of the heterogeneous capacity model is constant and also the homogeneous capacity model and both models have a negative slope and linear relationship



Algorithms	Homo	Hetro
MTPWBCR (β = 1)	537.3039	538.2452
MTPWBCR (β = 2)	537.1769	538.0881
MTPWBCR (β = 3)	537.0618	537.9427

Figure 4.8: Average Residual Capacity – Homo level 10 – (β = 1, 2, 3).

The Figure 4.9 displays the behavior of homogeneous and heterogeneous capacity models for different beta values. The behavior of the heterogeneous capacity model is constant as well the homogenous capacity model and both models have positive slope and linear relationship.



Algorithms	Homo	Hetro
MTPWBCR (β = 1)	0.26961	0.27548
MTPWBCR (β = 2)	0.28231	0.29119
MTPWBCR (β = 3)	0.29382	0.30573

Figure 4.9: Average Transmitted Power Homo – level 10 (β = 1, 2, 3).

4.3.5 Variance of Residual Capacity – Nodal Capacity Balance Indicator

We have looked at the average residual capacity for different algorithms in the previous sections. The variance is being calculated to get better understanding of the residual capacity at every node. The variance would show whether the residual capacity at every node at the network is balanced or not. Therefore the variance is being calculated for the residual battery capacity at every node. The calculation is being done to see whether the residual capacity is being balanced across all the nodes. The table below lists the variance for all the algorithms that were being simulated in this report. The results presented in the table shows that algorithms, MTPR and MTPWBCR are having a balanced residual capacity for both levels 1 and 10. At level 10, the algorithm, MTPWBCR is better than MTPR. The MBCR algorithm seems to have the unbalanced residual capacity at level 10.

Algorithms	Homo level-1	Homo level -10
MHCR	0.01280	1.28056
MTPR	0.00212	0.21285
MBCR	0.02828	1.38097
MWBCR	0.02828	1.12026
MTPWBCR	0.00212	0.13728

Table: 4.1 Variance for all the algorithms.

The variance is being shown for MBCR algorithm with different beta values at the table below. The residual battery capacity is similar for all the MBCR algorithms at level 1. As the beta value increases, it can be noticed that the residual capacity is balanced across the nodes.

Algorithms	Homo level-1	Homo level -10
MBCR	0.02828	1.38097
MWBCR($\beta = 2$)	0.02828	1.21433
MWBCR($\beta = 3$)	0.02828	1.12026

Table: 4.2 Variance for MBCR with $\beta = 1, 2, 3$.

4.4 Sample Source Code

Some important parts of the program are given below. These modules get modified appropriately depending on the performance matrix that we prefer to analyse. It also could be modified to suit an individual's needs.

```
public static void main( String [ ] args ) {
     GraphMBCR graph = new GraphMBCR();
     String sourceName = "";
     String destName = "";

   try{
        String line = "";
        FileReader fileReader = new
FileReader("c:/java/homo/sourceDestination.txt");
        BufferedReader bufferReader = new BufferedReader(fileReader);
        graph.clearFile();
        while(line != null){
        line = bufferReader.readLine();
                if (line != null)
                {
                StringTokenizer st = new StringTokenizer(line);
                sourceName = st.nextToken();
                destName = st.nextToken();
                 graph.MBCR(sourceName, destName);
                 graph.readInCapacity();

                for(int i=0; i < 4; i++){   //This loop could be changed for our needs
        and
                                        // it is for the number of packet
        transmission.
                        graph.readInResult();
                        graph.clearFile();
                        graph.setCapacity();
                        graph.setValues(sourceName, destName);
                        graph.setGraph(sourceName, destName);
                 }
                        graph.readInResult();
                         graph.setCapacity();
                         graph.setValues();
                         graph.calTotalResidualCapacity();
                         graph.clearFile();
                }
                else{
                        System.out.println("Reached EOF");
                }
        }
                        .
                        .
                        .
```

```
public void setGraph(String sourceName, String destName){
        GraphMBCR graph = new GraphMBCR();
        for(int i=0; i < vertex1.length; i++){
                double cost = 1/capacity[i];  //This cost function will take different
                                              //equation depending on the performance
                                              //metric
                graph.addEdge(vertex1[i], vertex2[i], cost);
        }
        graph.processRequestMBCR(sourceName, destName);
}

public void MBCR(String sourceName, String destName){
    FileReader fin;
    String fileName= "";
                        .
                        .
                        .

      while( ( line = graphFile.readLine( ) ) != null ) {
                StringTokenizer st = new StringTokenizer( line );
                try {
                        if( st.countTokens( ) != 4 ) throw new Exception( );
                        String source  = st.nextToken();
                        String dest    = st.nextToken();
                        double transmitPower = Double.parseDouble(st.nextToken());
                        double cost = Double.parseDouble( st.nextToken());

                        vertex1[index] = source;
                        vertex2[index] = dest;
                        power[index] = transmitPower;
                        capacity[index] = cost;

                        cost = 1/cost;  //This cost function equation will be different
                                        //depending on the performance metric
                        g.addEdge(source, dest, cost);
          }
          catch( Exception e ){
            System.err.println("Error: " + line);
          }
          index++;
      }
                        .
                        .
```

Chapter 5

Summary and Conclusions

This project focuses on the power aware algorithms for multi hop networks and measures their performance in terms of commonly used performance metrics. The report describes the multi hop network model as well as the parameters that contribute to the proper functionality of the model. It also presents a detailed analysis of the some of the popular power aware algorithms and explains them using numerical examples. The computer simulation of a sample network model was written using Java programming language as well as the results produced using the simulations are being discussed in detailed fashion in the previous section.

In this project we simulated a simple multi hop network to do the performance analysis for some important performance metrics such as delay, transmitted power and residual capacity. MHCR is having the lowest delay value for all the algorithms simulated and that was expected from theoretical point of view. MTPR is having the lowest transmitted power for all the algorithms simulated. MTPR and MTPWBCR are having the highest residual capacity for both the homogeneous and heterogeneous capacity models. This behaviour is observable due to the fact that MTPR and MTPWBCR use the transmitted power to find the shortest path, and the capacity is deducted by the transmitted power for all the algorithms. The summary of the results of the simulation is given in Table 5.1 as they are ranked 1 to 5 in terms of their performance for the homogeneous model of ten-packet transmission.

The variance for all the algorithms were also calculated using the simulation results. The variance is being calculated for the residual battery capacity at every node. The calculation is being performed to speculate whether the residual capacity is being balanced across all the nodes. The results for algorithms, MTPR and MTPWBCR show that these algorithms are having a balanced residual capacity for both levels 1 and 10. At level 10, the algorithm, MTPWBCR is better than MTPR. The MBCR algorithm has the unbalanced residual capacity at level 10.

Algorithms	**Delay**	**Transmitted Power**	**Residual Capacity**	**Variance**
MHCR	1	5	5	4
MTPR	4	1	1	2
MBCR	2	4	4	5
MWBCR	3	3	3	3
MTPWBCR	5	2	2	1

Table 5.1: Summary of Results.

Although the analysis of the currently available routing protocols for mobile multi hop network is presented in this report, their performance analysis is still a work in progress. There are some more performance metrics to consider in the simulation. The following parameters were not considered in this simulation: (1) Route Discovery Time (2) Communication Throughput (3) Packet Loss and (4) Route Reconstruction Time.

In order to perform the simulations, a simple mobile multi hop network would be programmed. This sample mobile multi hop network would then be simulated to locate the shortest path from the source node to the destination node as done by the routing protocols. The routing protocols then are simulated in the above-mentioned manner using the already created simple mobile multi hop network.

References

[1] C. Perkins, "Multi hop Networks", Addison-Wesley Publication Co., 2001.

[2] C-K. Toh, "Multi hop Mobile Wireless Networks", Prentice Hall, 2002.

[3] C.-K. Toh, "Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Multi hop Networks," IEEE Communications Magazine, pp. 138-147, June, 2001.

[4] K. Balasubramaniam and A. Anpalagan, "Performance Analysis Of Two Dynamic Link Selection Of Algorithms In Ad Hoc CDMA Networks", CD Proc. IEEE Canadian Conference on Electrical and Computer Engineering, May 2003

[5] Safwat , Ahmed et al, "Power–Aware Wireless Mobile Multi hop Networks" Handbook of Multi hop Wireless Networks, M. Ilyas (ed.), CRC Press, Dec. 2002.

[6] Sahni, Sartag, "Data Structures, Algorithms, and Applications in Java", McGraw-Hill Co., 2000.

[7] T. S. Rappaport, "Wireless Communications: Principles and Practice", 2nd Ed., Prentice Hall, 2002.

[8] http://people.westminstercollege.edu/faculty/ggagne/fall2002/306/labs/lab8/#weighted

[9] http://www.cs.fiu.edu/~weiss/cop3530_sum96/lectures.html

[10] http://www.csd.uwo.ca/courses/CS457a/reports/handin/asaadat/A2/routbasic.html

[11] http://www.pulsewan.com/data101/routing_basics.htm

Appendix

Description of Program: Routing Algorithm Simulation

Some of the important classes and methods are described in this section of the report.

The class Edge has only the constructor. This class represents the basic items in the adjacency list.
public Edge(int d, double c)

The Vertex class has only the constructor and represents the basic item stored for each vertex.
Vertex(String nm)

This following methods are in the Hashtable class. This class represents the basic entry in the vertex Hashtable.
public HashItem( ) */* Constructor */*
public HashItem(String nm) */* */*
public int hashCode() */* */*

The class Path implements the Comparable interface. The Objects are stored in the priority queue for Dijkstra's algorithm in this class.
class Path implements Comparable

The following is the important class, which has all the necessary methods in it to perform the Dijkstra's shortest path algorithm for weighted graphs. This class is implemented for the MBCR algorithm and it is called GraphMBCR.
class GraphMBCR

The high level descriptions of the methods within the GraphMBCR are as follows:
This is the main class. In this class all the source-destination are read in from the file sourceDestination.txt. For every source-destination pair the Dijkstra's shortest path algorithm is performed for the number of packet transmission specified in the loop. Also this prints all the global variables.
public static void main(String [ ] args)

This method clears all the contents in the outputToRead.txt file so that the next set of results could be put in for reading.
public void clearFile()

readInResults method reads all the vertices which were used in the routing from the outputToRead.txt file and stores them into a global array called results[].
public void readInResult()

This method reads in all the vertices along with their capacity from the graphCapacity.txt file and stores them into the global arrays called capacityVertex[] and singleCapacity[].

public void readInCapacity()

The capacity is deducted from the appropriate vertex in this method. For all the vertex which were used in the routing of every packet transmission the transmitted power is calculated. Then the transmitted power is deducted from the capacity of the appropriate nodes, which were used in the routing.

public void setCapacity()

This method calculates the total of the capacity remains in all the nodes when the packet transmission is completed.

public void calTotalResidualCapacity()

This method set all the global variables appropriately so that they could be used for our comparison. This method does not take any parameters.

public void setValues()

This method is same as setValues(), but this method takes the source-destination pair as the parameter so that the shortest path call could be made from this method.

public void setValues(String sourceName, String destName)

This method sets the graph with the appropriate transmission and residual capacity after every packet transmission.

public void setGraph(String sourceName, String destName)

In this method the initial graph information is read from the file graphPowerCapacity.txt. The information that was read from the file is used to set the graph and also the values with their corresponding vertices are stored into the global arrays for later use. This method is called in the main method to initiate the graph and perform the first shortest path on the graph.

public void MBCR(String sourceName, String destName)

This method finds the shortest path using Dijkstra's shortest path algorithm and prints the path on the output screen.

public boolean processRequestMBCR(String sourceName, String destName)

This is used to set the graph. The source and destination creates a link and cost represents the link weight.

public void addEdge(String source, String dest, double cost)

This method adds a vertex to the graph. If vertexName is an already seen vertex, return its internal number. Otherwise add it as a new vertex, and return its new internal number.

private int addNode(String vertexName)

Adds an edge to the graph for the given internal numbers of its vertices.

private void addInternalEdge(int source, int dest, double cost)

This method initializes the table.

private void clearData( )

This recursively prints the shortest path from the source node to destination node (specified by its internal number). This is the driver routine.
public void printPathRec(int destNode)

This is a driver routine to handle unreachables and to print total cost. It calls the recursive routine to print the shortest path from the source node to destination node after a shortest path algorithm has run.
private void printPath(int destNode)

This method computes the unweighted shortest path. This method is used to calculate the hop count algorithm
private void unweighted(int startNode)

This method implements the Dijkstra's Algorithm using binary heap. This returns false if negative edge is detected.
private boolean dijkstra(int startNode)

The class ReadWrite is used to write the network model into a text file with the links and their appropriate link weights. The main method is written to execute all the methods in the class. The writeSourceDestination method write 100 random source destination pairs in to a file called sourceDestination.txt. uniformCapacity method is used to select a random capacity between 10 and 20 and gets assigned to all the 36 nodes and stored into an array as well as it gets written into a file called graphCapacity.txt. The assignCapacity method assigns the capacity that got created by the method uniformCapacity to the appropriate link and writes that into a file along with the transmitted power. The method FileReadWrite reads the forward links and the reverse links between the nodes in the network from a file and assigns a random transmitted power and writes them into a file.

public class ReadWrite {
public static void main(String args[])
public void writeSourceDestination()
public void uniformCapacity()
public void assignCapacity()
public void FileReadWrite()
}

Data Structures (Support Files)

The following java files are needed to run the Dijkstra's shortest path algorithms. They contain all the necessary data structures of the program.

BinaryHeap.java
LinkList.java
LinkListItr.java
ListNode.java
QueueAr.java
ItemNotFound.java
UnderFlow.java