3-30-2004

A Control Layer Algorithm for Ad hoc Networks in Support of Urban Search and Rescue (USAR) Applications

Venkatesh Ramarathinam
University of South Florida

A Control Layer Algorithm for Ad hoc Networks in Support of Urban Search and Rescue (USAR) Applications

by

Venkatesh Ramarathinam

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Date of Approval:
March 30, 2004

Keywords: link stability, ad hoc routing protocols, energy efficiency, link breakage

ACKNOWLEDGEMENTS

I take this opportunity to express my sincere thanks to Dr. Miguel Labrador, for giving me this wonderful opportunity of working on this project. I am also grateful to him for his extended support and guidance throughout the course of this work, and for making my study at USF a pleasant and exciting educational experience. My sincere thanks to Dr. Katkoori and Dr. Valavanis, for being in my committee and for their valuable comments and suggestions.

It takes more than words to express my thanks to my family for their constant motivation and support, without which this work would not have been possible. I thank all my friends for their continuous encouragement and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

A CONTROL LAYER ALGORITHM FOR AD HOC NETWORKS IN SUPPORT OF URBAN SEARCH AND RESCUE (USAR) APPLICATIONS

Venkatesh Ramarathinam

ABSTRACT

Ad hoc networks have gained significant importance and gathered huge momentum within the wireless network research community. We explore the novel idea of applying ad hoc networking for urban search and rescue operations. Several algorithms have been proposed and implemented for routing in ad hoc networks and their performance have been thoroughly analyzed. But none of the prior work deals specifically for search and rescue operations, which entail certain specific criteria such as prevention of node loss, maximizing the area of coverage and constant and instantaneous access to a main controller.

In this thesis, we propose a centralized and adaptive algorithm tailored for efficient performance of mobile nodes assisting in search and rescue operations. The proposed algorithm assists in finding and maintaining stable links between the mobile nodes and base station, while optimizing the area of coverage and energy efficiency of the nodes. The algorithm is implemented using *ns* (network simulator), and its performance is compared with that of a widely used ad hoc routing protocol, Ad hoc On-demand Distance Vector (AODV) routing protocol. We use frequency of link breakages, network throughput and routing overhead as our performance metrics. This algorithm can also be extended to provide support for routing among mobile nodes.

CHAPTER 1

INTRODUCTION

In the recent past, we have witnessed a tremendous growth in deployment of wireless network technology driven by the need for ubiquitous service and rapid developments in telecommunications infrastructure. Mobile hosts such as notebook computers, featuring powerful CPUs and gigabytes of disk space are now easily affordable and becoming quite common in everyday life. At the same time, huge improvements have been made in wireless network hardware, and efforts are being made to integrate the two into a meaningful resource such as the Internet. We are witness to large scale proliferation of mobile computing and wireless technology in our day-to-day lives in the form of various hardware interfaces and technology devices, running numerous applications catering specifically to wireless technology. The use of cell phones and PDA's for mobile video conferencing, GPS based tracking systems and remote wireless sensor surveillance gives us an indication of the growth and proliferation of wireless technology in today's world.

The increased demand and usage of mobile devices, directly correlates to the inflated demand for mobile data and internet services. According to a study by Cahners In-Stat Group, the number of subscribers to wireless data services would reach 1.3 billion by end of 2004, and the number of wireless messages sent per month will reach 244 billion by December 2004 [1]. But these devices and technology use the standard wireless network model of a base station, repeaters, access points, and wireless nodes (Figure 1. 1). Oftentimes however mobile users will want to communicate in situations in which no fixed wired infrastructure is available, because it may not be possible to provide the necessary infrastructure or because the expediency of the situation does not permit this installation. The term ad hoc networks refer to such a collection of wireless mobile hosts forming a temporary network without the aid of any established infrastructure or centralized administration [2] (Figure 1. 2).

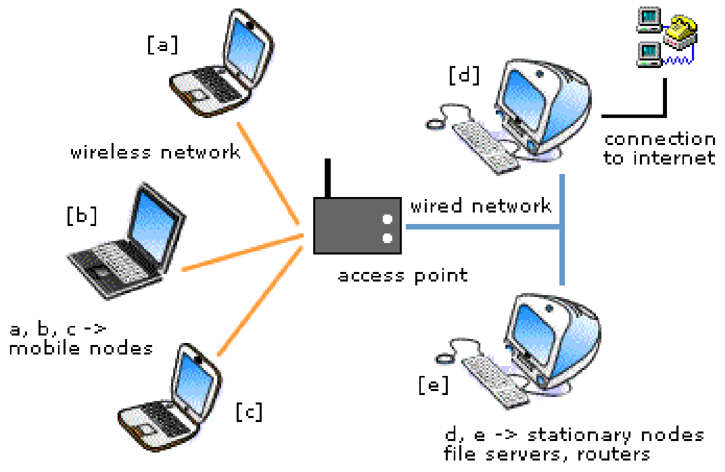

Figure 1. 1 Standard Wireless Model with Mobile Nodes and Fixed Infrastructure

1.1 Background

The history of ad hoc networks dates back to the DARPA radio packet network in 1972 [3], which was primarily inspired by the efficiency of the packet switching technology, such as bandwidth sharing and store and forward routing, and its possible application in mobile wireless environment [1]. But, it was not until the early 90's when research in the area of ad hoc networks gained significant momentum and widespread attention. This could be attributed to the surge in cheap availability of network hardware, micro computer revolution, and the increasing number of applications that required an ad hoc network kind of setup. MANET's or Mobile Ad hoc Networks have gained significant momentum as they are the solution for providing network services to mobile users at places where there is no infrastructure or an existing infrastructure needs wireless extensions. Some of the common applications of ad hoc networks are: conference halls, classrooms, search and rescue operations, vehicular communication, wireless surveillance and military operations.

In an ad hoc network, every node acts as a router, and forwards packets towards the destination. It is a self-organized network where every node cooperates to provide connectivity and services. Consider the topology shown in Figure 1. 3, with 6 mobile nodes forming an ad hoc network. Even though there is no direct link between node 1 and node 6, packets from node 1 are delivered to node 6 through intermediate nodes 2, 3, 4 and 5. Such an arrangement offers many advantages, some of which are listed below:

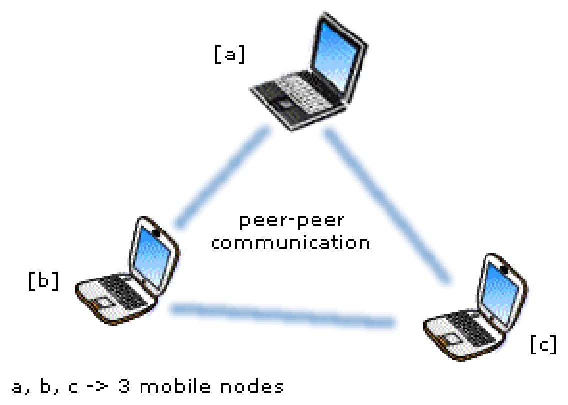

Figure 1. 2 An Ad Hoc Network of 3 Mobile Nodes

- Ad-hoc networks provide for fast and instantaneous set up, reducing the cost due to lack of fixed infrastructure.
- When applied to scenarios such as search and rescue operations ad hoc networks result in better performance than conventional wireless networks because of their non-hierarchical distributed control and management mechanisms [4].
- With proper power control mechanisms and relaying, they provide for inherent scalability and increased area of coverage, where each node adds to the existing network capacity.
- Provides for increased mobility and greater flexibility in network topology as an ad hoc network could be brought up and torn down in a very short time
- Ad hoc networks are well suited to use unlicensed bands and provide for increased spectrum reuse possibility.

Associated with these advantages and application possibilities are some inherent drawbacks that hold ad-hoc networks far from being deployed on large-scale commercial basis. Some fundamental ad-hoc networking problems that remain unsolved or need optimized solutions are listed below.

- The nodes in an ad hoc network can move arbitrarily which results in a very dynamic topology and frequent link breakages, disrupting communication between nodes.
- Often times nodes operating in an ad hoc network rely on battery for energy, thus for these nodes energy-efficient protocols become a critical design criterion.

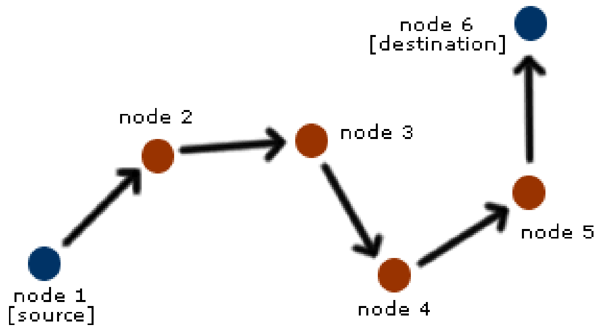

Figure 1. 3 Example Topology of 6 Mobile Nodes

- Bandwidth utilization is another significant factor for concern, thus necessitating reduced routing overhead and good congestion control mechanisms.
- Wireless networks are prone to physical and information security vulnerabilities, and is more dominant in ad hoc networks, where there is no control mechanism and centralized administration.

1.2 Urban Search And Rescue (USAR)

Traditionally there have been different fields of rescue operations[5]:

- Woodland/wild land search & rescue
- Mountain search and rescue
- Water search and rescue (includes diving)
- Urban search & rescue (includes confined space rescue)
- Combat search & rescue
- Cave search and rescue

Of the above mentioned fields of rescue operations, Urban Search and Rescue focusses on locating life and resources at man made structures, such as buildings collapsed in an earthquake, or at disaster sites affected by artificial or natural calamities. Recent research investigates the use of robots in such scenarios. These disaster sites pose several situational hazards that drastically affects the efficiency of human rescue teams. Some of the most common of these hazards and limitations are listed below [6]:

- Disaster sites are inherently unsafe and movement inside these sites is extremely restricted due to availability of only small or no entry voids, to explore the rubble.
- Vibrations due to mobility might further affect the foundation of the collapsed construction and could trigger a secondary collapse.
- Disaster sites are usually contaminated by damage to water/sewage distribution systems, toxic gas spill, body fluids and other hazardous materials and gases.
- Oftentimes, rescue operators would first need to extinguish the blazing fire in these sites before proceeding with any kind of rescue operations. This soaks the entire site and leaves it wet and slippery.

All the above mentioned factors coupled with the lack of enough trained skilled professionals makes it imperative to look for other effective means to carry out rescue operations. The use of mobile robots provides an effective alternative for improved efficiency in urban search and rescue operations. Due to smaller sizes and robust design, robots can explore disaster sites that pose numerous hazard threats and are not conducive for exploration by relief workers.

1.2.1 Communications in USAR Operations

Usually we have a team of autonomous mobile robots surveying a disaster site for life and resources[7]. Communication between these robots is critical to the performance in search and rescue operations. This would facilitate tele-operation of the robots and also provides vital information on the findings by the robots to the main controller. The IEEE 802.11 standard [8] for wireless LAN is currently used as the communication platform for these robots. But communication between the robots is heavily disrupted by interferences due to metal and other structural debris, and is also affected by the use of available radio channels by rescue personal and media [6]. Also, when operating autonomously, the mobile robots need to constantly communicate with the main controller situated near the disaster site. This severely affects the available area of mobility for the mobile robots. Any robot that has moved out of the transmission range of the main controller could be assumed to be trapped in the debris or to be lost. This severely affects the performance of these robots and also induces financial damage. Thus for better performance results and effective usage of robots for search and rescue operations, constant communication between the mobile robots and the main controller needs to be ensured, while also accounting for maximized area of coverage. By providing a constant communication link between the mobile robots and the main controller, it is ensured that the robots do not get lost; the term *node connectivity* is introduced here to denote the same.

Node connectivity is defined as the ability of a node to continue or stop its mobility without breaking away from the network of nodes, while remaining in constant communication with the main controller.

Forming an ad hoc network of the mobile robots and the main controller effectively addresses the issue of maximized area of coverage. By forming an ad hoc network, intermediate nodes act as a router forwarding packets towards the destination. By this method, robots continue their mobility beyond the transmission range of the main controller, if there exists an intermediate node through which it can establish a connection with the main controller. However, forming an ad hoc network of mobile robots does not address the issue of node connectivity. It is essential to ensure node connectivity in applications where loss of a node (mobile robot in the case of urban and search and rescue operations) could be detrimental to the performance of the system.

1.3 Motivation

Vast majority of the research work done in the area of ad hoc networks has been focussed on designing and developing routing protocols to address the issues of node mobility, overhead and energy efficiency. There has been an increased attention in developing routing protocols that consider the issue of link stability. [9] and [10], discuss design of link stability based routing protocol, where routes to destination are selected based on the strength of signals received from neighboring nodes or duration for which the link has been active. But none of the existing works guarantee link stability or provide node connectivity. Node connectivity is of utmost importance in scenarios where the mobility of the node is completely autonomous, or breakage of a node from a network is detrimental to the cost and performance of the system. Though the existing protocols suggest selection and use of stable links for communication, they do not guarantee link stability or account for node connectivity. This work addresses the issue of node connectivity in ad hoc networks, and provides a reliable solution for Urban Search and Rescue (USAR) applications, where the mobility of the node (robots) is stopped before breaking away from the network.

1.4 Contribution of this Thesis

This thesis investigates methods of providing stable links and constant node connectivity for mobile ad hoc networks. The main contributions of this work are:

- Provides a comprehensive review of the current and past work done in the area of routing protocols for mobile ad hoc networks and outlines the advantages and disadvantages of these protocols when applied to search and rescue operations.
- Introduces the concept of *control layer* for ad hoc networks, inter-operating between the routing and the data link layers, and suggests the design of a control layer algorithm to provide for node connectivity and link stability.
- Evaluates the performance of the proposed algorithm by comparing it with existing protocols using standard performance metrics.
- Implements the algorithm in the ns-2.26 simulator, and is made available for further research in this area.

1.5 Organization of this Document

This chapter provides a brief outline and motivation for this work. The next chapter provides an overview of the background related to this material, and reviews the past and current work relevant to this field. The third chapter discusses design considerations and explains the algorithm with the help of flowcharts and pseudo code, and make use of a sample model to illustrate the working of the algorithm. The fourth chapter describes the simulation setup and implementation details, analyzes results and evaluates the algorithm based on standard performance metrics and by comparing it with existing protocols. The final chapter concludes this work and enlists the scope for future work.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter reviews some of the well researched and published works on ad hoc routing protocols and discuss their suitability for application to urban search and rescue operations. It also provides a brief overview of the simulator and other tools used to implement the proposed algorithm.

2.1 Design of Ad Hoc Routing Protocols

Routing is a very critical aspect to the performance of an ad hoc network, and rightly, vast majority of the research in this area has focussed on designing efficient routing protocols. The design of a routing protocol for ad hoc networks needs to account for the highly dynamic network topology, battery power of the mobile nodes, minimized routing overhead and should provide stable links. Also, the design of the routing protocol needs to be decentralized, because in an ad hoc network every node performs the same functions, and there is no central administrator of the network.

2.1.1 Overview of Routing Methods

The problem of routing in an ad hoc network is similar to the distributed version of the classical shortest path problem [11]. Every node in the network maintains a list of preferred neighbors (next hop nodes) for each destination in the network in the form of a routing table. Every data packet contains a destination node identifier in its header. A node receiving a data packet checks the destination header field, and if it is not the intended destination, forwards the packet to the next hop neighbor towards the destination based on the information in its routing table. The design of the various ad hoc routing protocols differ in the manner in which the neighbor information, routing tables, and next hop towards destination are estimated, maintained and updated. Essentially, all routing protocols attempt in achieving the optimal path towards destination. Routing protocols also differ in the manner in which they estimate optimal path. Some protocols use distance between neighbors as a measure of determining the next hop towards a destination, while some protocols select the next hop based on the duration for which a link to that node has been active.

Routing protocols use the standard next hop routing methods and can be classified into the two primary categories of *link-state* and *distance-vector* routing. In a link-state approach each node in the network maintains a view of the current network topology, and a cost is associated to every link in the network. In order for all nodes to have a consistent view of the network, each node forwards its cost information of the *outgoing links* to all other nodes, using a protocol such as flooding. Upon receiving this information, the nodes update its view of network topology and applies a shortest path algorithm to select the next hop towards each destination. After the initial set-up time of the network, link cost updates are broadcasted only when there is a change in the network topology. This might introduce temporary stale routes, but it is usually corrected by the time it takes a message to traverse the diameter of the network [12]. Link state protocols provide very good scalability and the routing overhead is minimal even at higher node density. An example of link-state routing is the Open Shortest Path First (OSPF) [13] protocol.

Distance-vector routing uses hop count to determine the best path towards a destination. The calculated route would be optimal in terms of the number of hops required towards the destination, but most often the best route towards a destination need not necessarily be the one with shortest number of hops. Convergence time is the time it takes for all nodes in the network to make changes to its view of the network topology, and distance-vector routing achieves convergence by periodically sending its routing tables to all nodes in the network. In networks with large number of nodes, distance vector routing takes time to converge and also results in increased overhead and lower bandwidth utilization. *Routing Information Protocol* [14] is a classic example of distance-vector routing.

Routing protocols for ad hoc networks can be classified as proactive, reactive, and hybrid protocols (See Figure 2. 1). Table-driven or proactive routing protocols sends its routing table as periodic updates to its neighbors or to all nodes in the network. By sending periodic updates the protocol ensures that every node in the network has a picture of the current network topology, and eliminates the presence of stale routes. But one of the inherent drawbacks of this method is that it increases the routing overhead and therefore reduces the available bandwidth for data packets. The increase

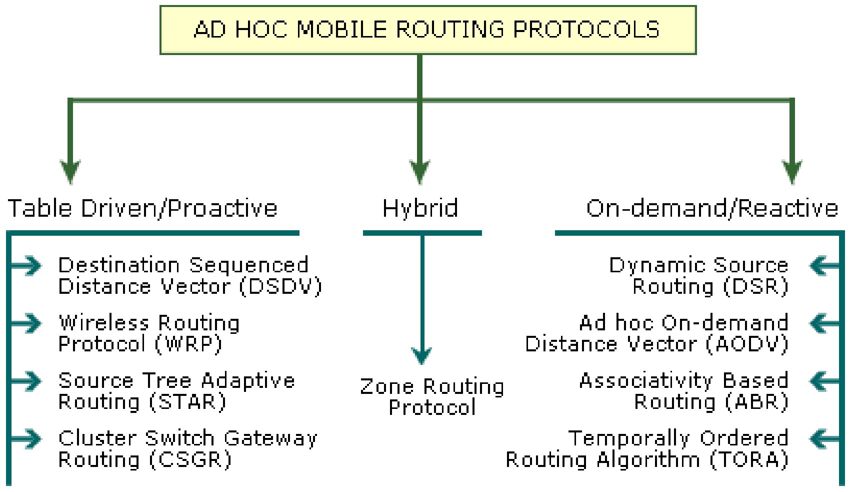
AD HOC MOBILE ROUTING PROTOCOLS
Table Driven/Proactive
Hybrid
On-demand/Reactive
Destination Sequenced Distance Vector (DSDV)
Wireless Routing Protocol (WRP)
Source Tree Adaptive Routing (STAR)
Cluster Switch Gateway Routing (CSGR)
Zone Routing Protocol
Dynamic Source Routing (DSR)
Ad hoc On-demand Distance Vector (AODV)
Associativity Based Routing (ABR)
Temporally Ordered Routing Algorithm (TORA)

Figure 2. 1 Table-driven, On-demand and Hybrid Ad Hoc Protocols

in routing overhead is directly proportional to the number of nodes in the network and to the rate of updates. Thus, the table driven approaches do not provide scalability and are also not apt for scenarios with little or no node movements. The different proactive protocols differ in the number of tables maintained, and the method by which changes in the network are broadcasted. Destination Sequenced Distance Vector (DSDV)[15] and Wireless Routing Protocol (WRP) [16] are examples of proactive or table driven approaches.

On-demand or reactive protocols, create routes only when required by the source node. When a node requires a route to a destination, it initiates a route discovery process within the network. All nodes in the network receive this packet, and either the destination node, or any other node that has route to this destination replies back to the source with the complete route information. Each node receiving this packet caches the route information and uses it for future packets towards this destination. This approach significantly reduces routing overhead, and negates the effect of routing loops. There are many ad hoc routing protocols in this category. Dynamic Source Routing (DSR)[2], Ad hoc On-demand Distance Vector (AODV) Routing [17] and Temporally Ordered Routing Algorithm (TORA)[18] are a few examples.

Hybrid routing protocols incorporate the merits of on-demand and proactive routing protocols. Nodes are grouped into clusters, and for nodes within the same cluster, a table-driven routing protocol is used, while to communicate with a destination node residing outside the cluster of source node, an on-demand search method is used.

2.2 Review of Ad hoc Routing Protocols

This section reviews some of the most common ad hoc routing protocols: Destination Sequenced Distance Vector (DSDV)[15], Dynamic Source Routing (DSR)[2], Ad hoc On-demand Distance Vector (AODV)[17], Temporally Ordered Routing Algorithm (TORA)[18], Associativity Based Routing (ABR)[19], Signal Stability-based Adaptive Routing (SSA)[10], and Location Aided Routing (LAR)[20]. Some of the other proposed protocols are discussed in [21, 22, 16, 23, 9, 24] and the references listed therein.

2.2.1 Destination Sequenced Distance Vector (DSDV)

The Destination Sequenced Distance Vector (DSDV) routing protocol was introduced in [15]. DSDV is a distance vector routing protocol that implements modifications to the Bellman-Ford routing mechanism as specified by [14], to make it suitable for a dynamic and self-starting network mechanisms. Every node in the network maintains a routing table with entries for every destination in the network along with the corresponding number of hops towards the destination. Also each entry is marked by a sequence number distributed by the destination node. The sequence numbers could be used to distinguish stale routes from new routes, thereby avoiding the formation of routing loops. DSDV requires each node to advertise to its neighbors its whole routing table, including the next hop information to reach all other destinations in the network. DSDV sends updates periodically (proactive) or in the event of a link change and uses sequence numbers in order to use the most recent information. Moreover the updates sent out might be fluctuating, and would create temporary routing loops. To dampen fluctuations, settling time is calculated for every node in the routing table, and updates are sent out after this settling time is reached. DSDV can be beneficially applied to ad hoc networks and guarantees loop free paths to each destination at all instants.

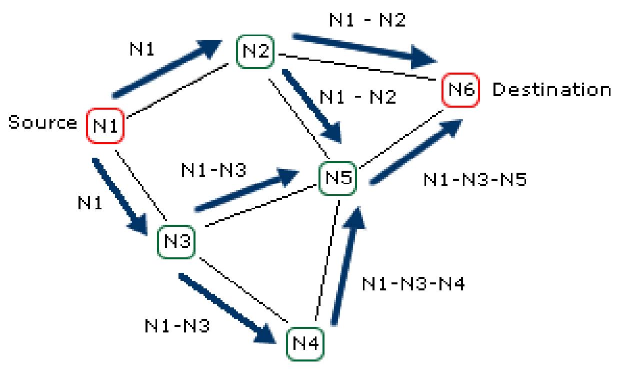

Figure 2. 2 Dynamic Source Routing (DSR) Method of ROUTE REQUEST

2.2.2 Dynamic Source Routing (DSR)

Dynamic Source Routing (DSR), presented in [2] is an on-demand (reactive) routing protocol that is based on source routing, where packets leave a source node towards the destination, with the complete route information in their headers. Each node maintains a local cache of source routes of which it is aware. When a node has to send a packet to a destination, it first checks it local cache for a source route for that destination. If it has a source route, it will use it to send the packet towards the destination. If there is no entry for the destination in the route cache, then the source node initiates a *ROUTE REQUEST* to that node. Each node receiving this packet appends its information in the *ROUTE RECORD* of the *REQUEST* packet. If the receiving node has a route to the intended destination, it appends this information to the *ROUTE RECORD* and forwards the packet to the node that initiated the route discovery. However, if the receiving node does not have a route to the destination, it broadcasts the *ROUTE REQUEST* packet to all nodes in its outgoing links. To avoid multiple broadcast of the same packet, a node forwards a packet only if it has not seen it already and its information is not appended to the *ROUTE REQUEST*. Figure 2. 2 illustrates the formation of route record as the route request message propagates through the network. When the *ROUTE REQUEST* is received by the destination node or an intermediate node with route to that destination, it generates a *ROUTE REPLY* with the reversed information from the route record and sends it towards the destination. Figure 2. 3 illustrates the route reply method of DSR. For route maintenance, DSR makes use of *ROUTE ERROR* packets to inform nodes in the network about a

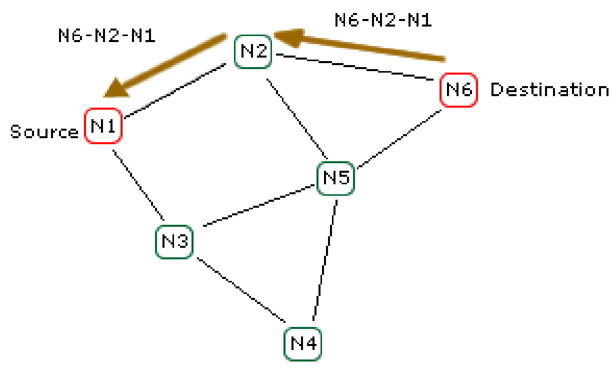

Figure 2. 3 Dynamic Source Routing (DSR) Method of ROUTE REPLY

stale route or broken link, and upon receiving a route error packet all nodes update the information in their route cache to reflect the current network topology.

2.2.3 Ad hoc On-demand Distance Vector (AODV)

Ad hoc On-demand Distance Vector (AODV) discussed in [17] is a reactive protocol based on DSDV and DSR. It borrows the idea of on-demand routing from DSR and uses the mechanism of hop-by-hop routing and sequence numbers from DSDV. Each node in the network maintains a routing table with entries for every other node in the network along with the sequence number of the route reply received from that node. A node **S** that requires a route to destination **D**, initiates a route discovery and broadcasts a *ROUTE REQUEST* packet to its neighbors, which in turn broadcast the request towards the destination. Unlike DSR, where each node receiving the route request appends its information to the route record, nodes receiving an AODV route request, initiates a *ROUTE REPLY* if it is the destination **D**, or has a route to the destination. Else, the node does a re-broadcast of the route request. Each node that forwards the *ROUTE REQUEST* creates a *reverse route* for itself back to node **S**.

ROUTE REPLY contains the number of hops towards the destination and the sequence number for the destination, most recently seen by the node generating the reply. Each node that participates in forwarding this reply back to the source or initiator of the *ROUTE REQUEST* (node **S**) creates a forward route to the destination **D**. Thus AODV uses hop-by-hop routing where each node re-members only then next hop towards a destination and not the complete route to a destination, as would be the case with source routing. Route maintenance in AODV is similar to that of DSR and is achieved by periodic broadcast of hello messages or through link layer detection.

2.2.4 Temporally Ordered Routing Algorithm (TORA)

Temporally Ordered Routing Algorithm (TORA)[18] uses a controlled flooding mechanism to discover multiple routes from a source to a destination on a demand basis. In order to reduce overhead, TORA sometimes utilizes sub-optimal paths instead of triggering a new flooding procedure. It is a loop-free distributed routing algorithm based on the concept of link-reversal. The main design idea of TORA is localization of control messages to a very small set of nodes near the occurrence of a topological change. TORA uses relative heights of node as a metric for path estimation, and each node views the network as a *directed acyclic graph* (DAG). When a node requires a route to a destination it broadcasts a QUERY packet containing the address of the destination for which it requires a route. This packet propagates through the network until it reaches the destination node or any other node that has a route to this destination. The receiving node then propagates a UPDATE packet with its height set relative to the destination. Each forwarding node then sets its height greater than the height of the neighbor from which it received the UPDATE packet. In this manner a series of directed links is created from the source to the destination. When a node detects a broken link to one of its previous neighbors, it sets it height so that it is at a local maximum with respect to its neighbors and transmits an UPDATE packet with this information. Because TORA uses inter-nodal coordination for route maintenance, there is a potential for temporary routing loops and fluctuations, however these are just temporary and route convergence would eventually occur.

2.2.5 Associativity Based Routing (ABR)

Associativity based Routing (ABR)[19] is a source initiated on-demand routing protocol. Associativity is a measure of the nodes connectivity with its neighbors. Nodes that have greater associativity have been stable for a longer duration, while nodes with lower associativity indicate greater mobility rates. Each node in the network broadcasts periodic beacons, and every neighbor receiving this beacon keeps a count of the frequency at which these beacons have been received. Large number of beacons exchanged between nodes indicate that the link has been stable for a longer duration.

A suggested modification of ABR makes use of battery power life of the nodes, signal strength and route relaying load in estimation of optimal path towards destination. There are three phases of the ABR protocol: (a) *route discovery* phase, (b) *route reconstruction* phase, and (c) *route deletion* phase. ABR lays emphasis on longevity of routes and significantly differs from the traditional distance vector routing algorithms that concentrate on estimating routes based on shortest path towards destination.

2.2.6 Signal Stability-based Adaptive Routing (SSA)

Signal Stability-based routing [10] uses signal strength and stability of the individual hosts as route selection criteria. Selecting the most stable links, links that exhibit strongest signals for maximum amount of time, leads to longer lived routes and lesser maintenance. SSA follows the on-demand routing paradigm. It differs from Associativity Based Routing (ABR) in regard to its route discovery mechanism. Unlike ABR, where routes are selected purely based on links that are stable for longer duration, SSA uses link stability coupled with signal stability. The authors claim that selection of route based only on link stability may not always yield better performance, due to the fact that nodes are highly mobile in an ad hoc network and has unpredictable mobility patterns.

In Signal Stability-based Adaptive routing, source nodes initiates a route discovery when it has data to send to a destination that is not in the routing table. Intermediate nodes forward the packets only if it is received over a *strong channel* and has not been forwarded earlier. Forwarded packets have the host information appended to its headers and the source node gets the complete route when it receives a route reply from the destination. Functionally, the SSA protocol consists of two protocols - Forwarding Protocol (FP) and Dynamic Routing Protocol (DRP). DRP makes extensive use of the underlying device driver interface, which forwards the signal strength information to the routing protocol. Unlike conventional routing protocols, two tables are maintained in SSA: the routing table and signal stability table. DRP maintains these tables and FP does the work of lookup and forwarding the packets on the appropriate channels.

2.2.7 Location Aided Routing (LAR)

While various routing protocols use different parameters such as number of hops, bandwidth of a link and signal strength of a link to estimate route information, Location Aided Routing (LAR)[20] utilizes location information to improve performance of ad hoc wireless networks. This clearly implies that the nodes need to have a method of estimating their current location, say via GPS. This assumption could severely handicap the application of LAR to scenarios where it might not be practically feasible to gather location information. LAR also makes another assumption that a sender node is aware of the destination location and velocity. In addition, without considering signal strength, battery power life, and connectivity information, communication performance will suffer if data are routed based on location information alone [25].

2.3 Evaluating Performance and Suitability Issues

A vast deal of research has also been focussed on analyzing the performance of the proposed protocols using different performance metrics and various scenarios. Some of the most common performance metrics are throughput of the network, packet delivery ratio, route establishment delay, convergence time, routing overhead and link stability. Also these metrics are evaluated for different scenarios such as various mobility rates, random pause times and under different load conditions. Comprehensive analysis have also been done to study the performance of TCP and UDP over these protocols. [26, 27, 28, 29] provides a performance review of AODV, DSDV, DSR and TORA under different settings. [30, 31] reviews some of the position based routing protocols. In [32] the authors have done a complete analysis considering various network possibilities and performance metrics, on the performance of TCP over ad hoc networks. [33] also includes a performance comparison of TCP over ad hoc routing protocols, but analyzes the performance over static ad hoc networks. [34] compares the performance of hybrid vs. proactive ad hoc routing protocols, while [35] reviews the performance of on-demand vs. table-driven routing protocols. However one of the important conclusions from the analysis done in [26] that influences the design choice is that for slower mobility rates, of the order of 1 meter/second, performance of AODV and DSR are similar and are better compared to other routing protocols.

Currently, communication between a group of autonomous mobile robots is achieved by forming a wireless LAN of the robots [36]. This model of communication is based on the *IEEE 802.11* wireless standard. The idea of forming a wireless LAN of mobile robots when applied to urban search and rescue operations has several drawbacks. Some of the reasons attributed to the same are listed below.

- Setting up a wireless LAN requires infrastructure (access point) and power to power up the access point, which are not readily available at disaster sites.
- Oftentimes, in USAR operations, the mobile robots maneuvering the disaster site would need to maintain constant communication with a stationary controller, transmitting search findings and location information. The main controller is usually stationary and provides scope for tele-operation and analyzes findings of the robots to provide meaningful information to the relief workers. To ensure this constant communication with the main controller, the mobile robots need to stay within the transmission range of the main controller. This severely affects the area of coverage and the performance of the system.
- Forming a wireless LAN of mobile robots requires an initial setup time
- Node connectivity is another important aspect in USAR operations. Nodes moving away from the transmission range of the main controller are considered lost, unless they use inherent position awareness protocols to trace route back to the main controller. Loss of a robot could induce financial damage as well as performance degradation. A network of mobile robots forming a wireless LAN using IEEE 802.11 standards does not provide for node connectivity.

However, most of the above mentioned issues could be resolved by forming an ad hoc network of mobile robots, instead of forming a wireless LAN. The idea of applying an ad hoc networking to a team of mobile robots is discussed in [37]. [38] reports the implementation of a link state routing protocol in wireless modems for application in mobile robot-based network. By forming an ad hoc network of mobile robots, the area of coverage is drastically increased, as it allows multi-hop routing, and robots can maintain communication with the main controller through intermediate nodes. Also there is no setup time or need for any fixed infrastructure (access point). Forming an ad hoc network also results in reduced network congestion as routing remains distributed. Further, the use of multi-hop routing provides many alternate routes for communication with the main controller.

Though forming an ad hoc network of mobile robots seem to be a plausible solution, none of the existing routing ad hoc protocols address the issue of node connectivity. For a wireless ad hoc network with a highly dynamic network topology, the use of optimal path routing algorithms would not yield good results as the routes are highly volatile and the network topology at an instant τ is not the same as the topology at $\tau + 1$. Shortest path routing is not of much use when the links are highly unstable. Rather, routing decision needs to be based on signal strength between nodes, duration of active links and the battery power life of the nodes. SSA[10] and ABR[19] ad hoc protocols addresses these issues. Though these protocols suggest routing data through stable links (SSA uses signal stability and duration of link as routing metrics, while ABR makes routing decisions based on link stability or duration for which a link has been active), they do not guarantee node connectivity.

This issue of providing/guaranteeing node connectivity, while maintaining the other benefits from implementing an ad hoc network is addressed in this work. Instead of creating an entirely new routing protocol to address this issue, the proposed solution takes advantage of the the researched work and suggests a *control layer* algorithm to ensure node connectivity. The idea of a control layer-based approach, choice of routing layer protocol, and the algorithm are explained in detail in the next chapter.

2.4 Simulator and Other Tools Used

The proposed algorithm is implemented using the *Network Simulator - ns* (version 2.26). *ns* is a discrete event simulator developed at the Lawrence Berkeley National Laboratory (LBNL) [39] with extensions from the MONARCH project at Carnegie Mellon [26]. The current implementation of *ns* supports simulation of real-time wired and wireless networks. It is a freely distributed, open source tool, written in C++, and uses OTcl as a command and configuration interface. *ns* provides a common platform for simulating/comparing different network models. It provides support for FreeBSD, Linux, Solaris, Windows and Mac platforms. For detailed information about this simulator the reader is referred to [39] and [26]. *ns* currently includes extensions for the most common ad hoc routing protocols, 802.11 MAC data link layer, and also free-space and two ray ground models for the physical layer.

In addition, the Network Animator (NAM) is used for visualization and debugging purposes. NAM is a Tcl/TK based animation tool for viewing network simulation traces and real world packet traces. It supports topology layout, packet level animation, and various data inspection tools. The NAM development effort was an ongoing collaboration with the Virtual InterNetwork Testbed (VINT) project. Currently, it is being developed at the Information Sciences Institute (ISI) at the University of Southern California (USC), as part of the SAMAN and Conser projects. NAM is a visualization tool used to read large files of data sets and be extensible enough so that it could be used to view different network situations. *ns* generates a NAM readable trace file format to view an animation of the simulated experiments. More information on NAM can be found in [40].

CHAPTER 3

PROPOSED ALGORITHM - DESIGN AND DEVELOPMENT

In this chapter the design of the algorithm is discussed along with an elaboration of the parameters and functional specifications. It also explains the algorithm with the help of a sample model.

3.1 Design Considerations and Choices for the New Algorithm

In Section 2.2 some of the most common and popular routing protocols for ad hoc networks were reviewed [17][15][2][10][20] and [41]. Though these address the basic issues surrounding an ad hoc network, they do not serve as the right model for robot assisted search and rescue operations which entail an added dimension of functionality - *NODE CONNECTIVITY*. The designed protocol makes sure that the nodes (mobile robots) do not get lost and a connection with the main controller always exists. Let us take the example of the scenario depicted in Figure 3. 1. There is a stationary main controller, and 6 mobile nodes (robots). Robots 1, 2, 3 and 4 are within the transmission range of the main controller (denoted by a circle), while robots 5 and 6 are outside its transmission range. This doesn't necessarily mean that robots 5 and 6 have lost their communication with main controller. For example, robot 6 can still communicate with the main controller through robot 2. Here robot 2 acts as a router, transmitting packets to and from robot 6. Similarly robot 5 can transmit its packets to the main controller through robot 3 or 4. But as it can be seen, robot 2 is moving outside the transmission range of the main controller. This not only breaks the communication link between robot 2 and the main controller, but also the link between robot 6 and the main controller, as robot 2 was serving as the link between these two nodes. None of the existing routing protocols address this issue as discussed in Section 1.3.

The new design of the routing protocol has to ensure that in addition to the existing demands of ad hoc networks such as node mobility, link stability, energy efficiency and reduced routing overhead, it had to provide for *NODE CONNECTIVITY*. Based on the information from this protocol,

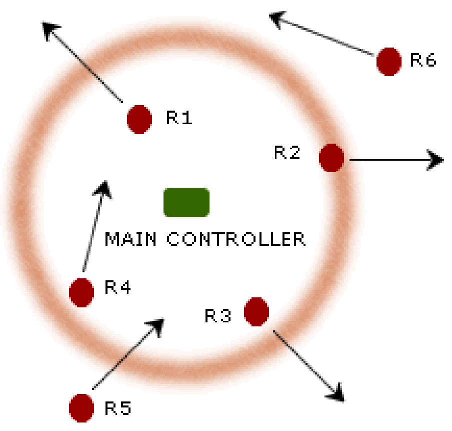

Figure 3. 1 Scenario Consisting of Main Controller and 6 Mobile Nodes

a node should stop its mobility when its connection with the main controller is in danger of being lost, which is indicated by a threshold based on signal strength and batter power of the nodes. Alternatively, an algorithm operating between the routing and data link layer could be designed, as represented in Figure 3. 2. This idea of having an inter-operating layer to provide for node connectivity, allows the use of existing MAC layer and routing layer protocols. In this case, the IEEE 802.11 standard provides for an efficient MAC layer protocol and the Ad hoc On-demand Distance Vector (AODV) acts as the routing layer protocol. An explanation for the selection of AODV as the routing layer protocol is provided in Section 3.3.

3.2 Approach and Design of the Algorithm

3.2.1 Outline of the Problem

As detailed in Section 1.3, node connectivity is critical to the performance in search and rescue operations using robots. Loss of robots are detrimental to both cost and effort, alike. Figure 3. 3 provides an insight to the problem. To provide for node connectivity, the design of the algorithm needs to ensure constant communication link between the mobile nodes and the main controller.

In addition to maintaining constant link between the robots and the main controller, the design of new algorithm should also ensure the following:

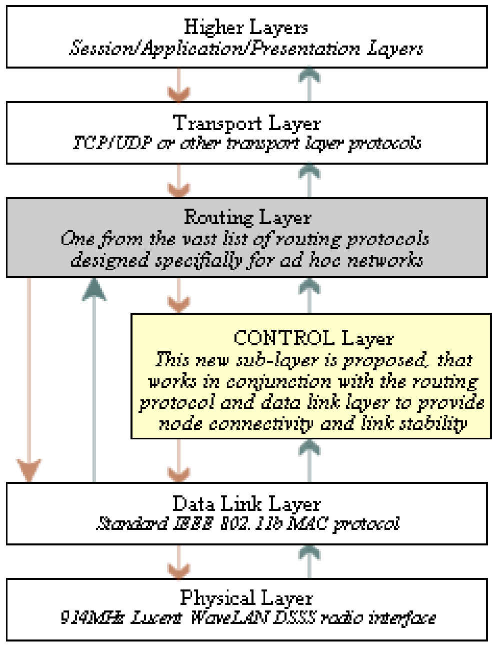
Higher Layers
Session/Application/Presentation Layers
Transport Layer
TCP/UDP or other transport layer protocols
Routing Layer
One from the vast list of routing protocols designed specifially for ad hoc networks
CONTROL Layer
This new sub-layer is proposed, that works in conjunction with the routing protocol and data link layer to provide node connectivity and link stability
Data Link Layer
Standard IEEE 802.11b MAC protocol
Physical Layer
914MHz Lucent WaveLAN DSSS radio interface

Figure 3. 2 Layered Approach for Design of the Protocol

- Increase in routing overhead should be minimal, as increased overhead leads to congestion in the network and wastage of battery power
- Area of coverage should not be confined and restricted for providing node connectivity
- Energy of the mobile nodes should be considered, as a node might be well within the transmission range of the main controller, yet have very little or no battery life. Such nodes should not be used to relay packets and should return to the main controller.
- Nodes at threshold of a connection should stop their mobility if breaking away from this connection would disrupt its communication with the main controller. For example, robot 2 in Figure 3. 1 is at threshold, as its direction of movement is away from the main controller. Its mobility would break its connection with the main controller as well as the connection of robot 6 with main controller, since robot 2 connects robot 6.
- Nodes should constantly monitor the signal strength of the packets received from its neighbors.

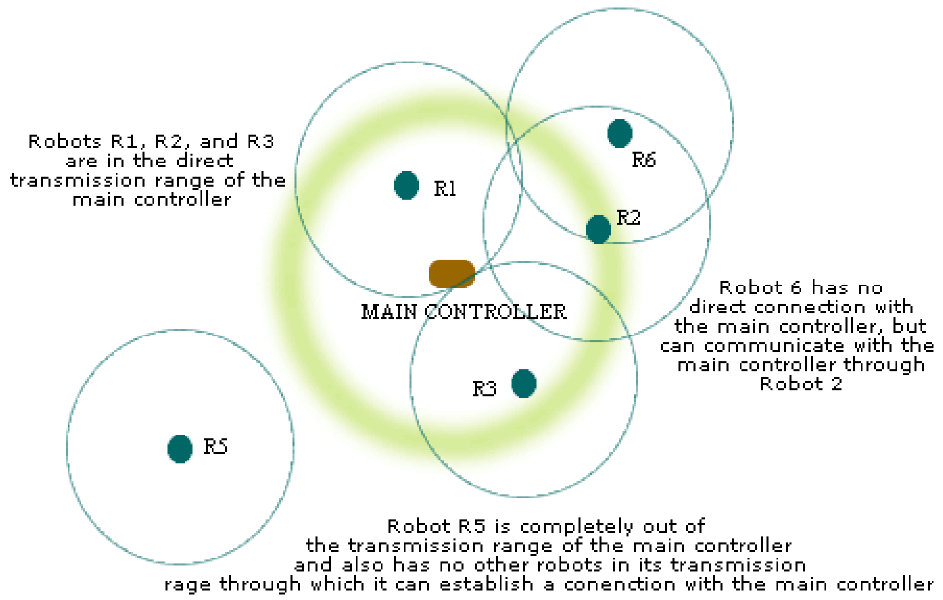
Robots R1, R2, and R3 are in the direct transmission range of the main controller
R1
R6
R2
MAIN CONTROLLER
Robot 6 has no direct connection with the main controller, but can communicate with the main controller through Robot 2
R3
R5
Robot R5 is completely out of the transmission range of the main controller and also has no other robots in its transmission rage through which it can establish a conenction with the main controller

Figure 3. 3 Sample Scenario of 6 Nodes with their Transmission Range

- If after stopping its mobility, a node is able to establish a link to the main controller through another node that has just moved within its transmission range, it should start its mobility and continue to move in the same direction at the time of it stopping its mobility, unless there is a tele-operated change in direction.

The designed algorithm, completely implemented in the control layer shown in Figure 3. 2, considers all the above listed required functionalities. The main functions and assumptions made in the design of the proposed algorithm are next summarized:

- The robots and main controller are deployed at the same location at the start of search and rescue operation. The main controller remains stationary throughout the entire operation and the robots start exploring the disaster zone.

- Every node broadcasts [1] periodic hello messages with its energy information in a standard packet format. The hello messages are exchanged once every *HELLO_INTERVAL*[2].

[1]A node receiving a broadcast message does not forward the packet, thus the packet is delivered only to the nodes that are within the transmission radius of the sender node

[2]HELLO INTERVAL is the time interval between successive hello messages broadcasted by the individual nodes

Table 3.1 Format of an Information Table Maintained at Individual Nodes

Neighbor ID	Normalized Energy level	Normalized Signal Power

Table 3.2 Format of an Update Table Maintained at the Main Controller

Neighbor ID	Composite Threshold

- Each node that receives the hello messages from its neighbors, calculates the signal strength at which the packet was received.

- An *information table* (see Table 3.1) is maintained at every node, where, for each node in the network an entry is maintained. This table has fields corresponding to the neighboring node id, its normalized energy level, and the normalized power level at which the hello packet was received from that node.

- At periodic intervals every node in the network, computes the *composite threshold*. for each of its neighboring nodes, based on the values in the information table. This value of composite threshold is stored along with the corresponding neighbor id in the *update table* (see Table 3.2). Each node then forwards its *update table* to the main controller.

  Composite threshold refers to a combined value of energy and signal strength calculated using the equation $((alpha * normalized\_energylevel) + (1 - alpha) * normalized\_powerlevel)$, *where the value of* $alpha$ *is dynamically estimated (Section 3.2.4).*

- Main controller receives update tables from all nodes at periodic intervals and performs a local computation on each of the received tables to see if every node in the network has a connection to the main controller.

- The main controller loops through every node in the network to check if it has a direct connection with the main controller or through any other intermediate node, or if it has such a connection, but is at the *link threshold* [3]. The main controller sends a message to these nodes, with its mobility flag set to *false* (Section 3.2.2).

[3]Link threshold is the minimum composite threshold for the network, below which the link is considered a weak link.

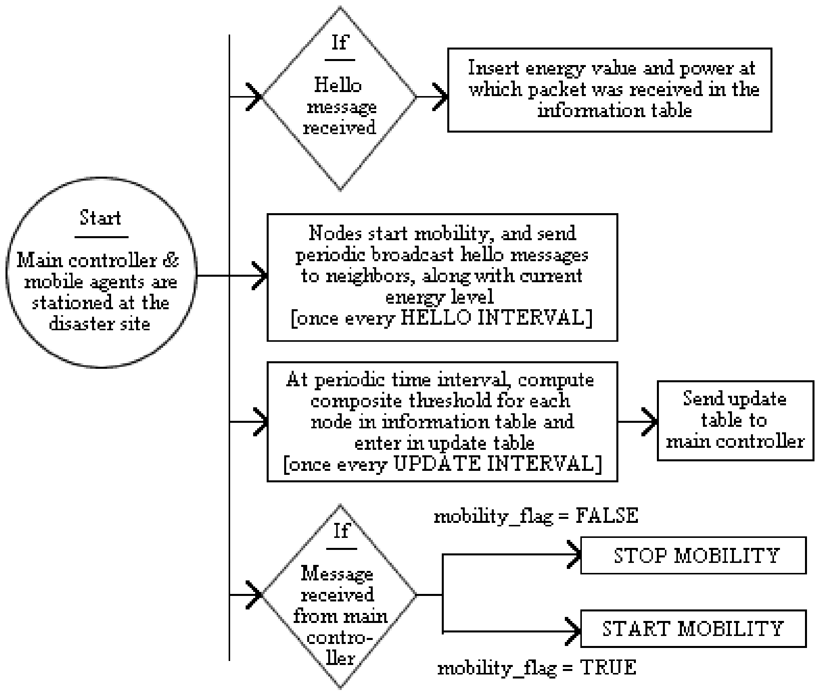
Start
Main controller & mobile agents are stationed at the disaster site
If
Hello message received
Insert energy value and power at which packet was received in the information table
Nodes start mobility, and send periodic broadcast hello messages to neighbors, along with current energy level [once every HELLO INTERVAL]
At periodic time interval, compute composite threshold for each node in information table and enter in update table [once every UPDATE INTERVAL]
Send update table to main controller
If
Message received from main contro-ller
mobility_flag = FALSE
STOP MOBILITY
START MOBILITY
mobility_flag = TRUE

Figure 3. 4 Flowchart for the Logic at Local Nodes

- Nodes that receive messages from main controller with its mobility flag set to false, stop their mobility (if not stopped already), and wait for a certain period of time to see if it receives a message from the main controller with mobility flag set to true, in which case it continues its mobility. If this wait time expires, and the nodes do not receive a message from the main controller with mobility flag set to true, they begin to move towards the main controller as a preemptive measure.

The designed control layer algorithm, uses a *centralized mechanism* to monitor mobility of the nodes, and all nodes use the underlying distributed ad hoc routing protocol to exchange hello packets and send the computed information table to the main controller. However, it is the main controller that decides on mobility of all the nodes based on the data in the information table, and hence this approach is classified as centralized. But this assumption could be easily justified. In the case of USAR applications, all the mobile robots need to have a constant communication link with the main controller at all times, thus making it a centralized mode of operation. The possibility of leaving the control of deciding on mobility to the local nodes, as in a distributed mechanism is discussed in

Section 3.4. Figure 3. 4 explains the logic at the individual nodes in the form of a flowchart. The algorithm at the main controller is independent of the logic at the individual mobile nodes and is discussed in the next section.

3.2.2 Algorithm at the Main Controller

The main controller receives update tables from every node in the network once every *UPDATE_INTERVAL*[4]. The information in these tables is copied into a data structure maintained at the main controller. A sample format of this data structure is given below:

```
struct BASESTATION_OBJECT {
        int number_of_neighbors;
        bool mobility_FLAG;
        double neighbor_id[n];
        double composite_threshold[n];
        long int update_seqno;
} instances[n]
```

As it can be seen from the above format, n instance of the above structure is created at the main controller, where n is the number of nodes in the network. Thus for each node in the network the main controller maintains a data structure comprising of number of neighbors of that node, mobility flag that specifies if that node is mobile or not, an array of neighbor id's, composite threshold values and the sequence number of the update packet expected from that node. Use of sequence numbers for update packets is similar to the implementation of sequence numbers for TCP packets. Entries in this data structure are modified as and when update packets are received.

The main controller runs an algorithm once every *MONITOR_INTERVAL*[5] to check for mobility status of every node in the network. The algorithm loops through the information table of every node in the network, and checks for a connection to the main controller that is above the composite threshold. If no such link exists, then it finds the neighbor of this node that has the maximum value of composite threshold and looks in the information table of the neighbor node for a connection (The node that has the maximum value of composite threshold among all neighbors, should have its composite threshold greater than the link threshold). The algorithm stops the mobility of the mobile

[4]UPDATE INTERVAL is the time interval between successive update packets sent by the individual nodes

[5]MONITOR INTERVAL is the time interval between successive loops through the algorithm at the main controller to check for mobility status of the nodes in the network

node, if after recursively iterating through the information table of all neighbors, it does not find a direct or multi-hop connection to the main controller. The complete algorithm is presented below.

```
nodes_visited = NULL;
for all node in the network
    for each neighbor of this node
        if ((neighbor == main controller) &&
                            (composite_threshold[neighbor] > link_threshold)) {
          continue_mobility(node);
        }
        else {
                parent = findparent(node);
                nodes_visited = nodes_visited + node;
                mobility = check_connection(parent);
                if (mobility == TRUE)
                            continue_mobility(node);
                else
                            stop_mobility(node);
    }


check_connection(node_id) {
    for each neighbor of this node
        if ((neighbor == main controller) &&
                            (composite_threshold[neighbor] > link_threshold)) {
             return TRUE;
        }
        else {
                parent = findparent(node);
                nodes_visited = nodes_visited + node;
                mobility = check_connection(parent);
        }
        if (mobility == TRUE)
             return TRUE;
        else
             return FALSE;
}


findparent(node) {
    select (PARENT ∈ neighbors of node) such that
        composite_threshold[PARENT] == max(composite_threshold ∀ neighbors)
        && composite_threshold[PARENT] >= link_threshold
        && PARENT ∋ nodes_visited
    if (no such node exists)
        return NULL;
    else
        return PARENT;
}


stop_mobility(node) {
    if (mobility_FLAG[node] == TRUE)
        set mobility_FLAG[node] = FALSE;
        send_message(node);
}


start_mobility(node) {
    if (mobility_FLAG[node] == FALSE)
        set mobility_FLAG[node] = TRUE;
        send_message(node);
}
```

If there is no direct link between any of the nodes and the main controller, the algorithm recursively loops through every *safe neighbor*[6] to find a link to the main controller. This ensure maximum *safe area of coverage* without loosing contact with the main controller. By making sure that all nodes have a direct or routed link to the main controller, the algorithm also ensure that there is a communication link between individual nodes, i.e. a tree structure of the network is always maintained.

3.2.3 Logic at the Individual Nodes

The algorithm at the mobile nodes has been represented by a flowchart in Figure 3. 4. The logic at the individual nodes relates to triggering discrete events and regular intervals of time. The algorithm works as follows.

- Each node broadcasts periodic hello messages, along with the local nodes battery power, once every HELLO INTERVAL.
- On receiving a hello message from one of its neighbors, it estimates the signal power at which the packet was received, and stores this information along with the battery power level of the neighbor from the hello message, in the local information table.

[6]Safe neighbors are neighbor nodes that have a composite_threshold value greater than link threshold

- Once every UPDATE INTERVAL, nodes calculate composite threshold for all entries in the information table and generate the UPDATE TABLE. This update table is then forwarded to the main controller.ALPHA value is estimated at the individual nodes (refer Section 3.2.4).

```
for each HELLO_INTERVAL{
        send_hello_packets();
}

for each UPDATE_INTERVAL {
        $\forall$($\in$ of information_table)
                estimate_composite_threshold();
        generate_update_table();
        forward_update_table_to_basestation();
        estimate_ALPHA();
}

for each packet_received {
        if (type==hello) {
                neighbor = packet.node_id;
                power = packet.recd_power();
                energy = packet.energy;
                insert data in information_table;
        }
        else if (type==basestation_message) {
                if (mobility_FLAG==STOP && node is mobile)
                        stop_node();
                else if (mobility_FLAG==START && node is stopped)
                        start_node();
        }
}
```

Function call *send_hello_packets()* sends a broadcast hello message to all neighbor nodes. As the name implies function call *generate_update_table()* estimates the composite threshold value for neighbor nodes in the information table and stores it in the update table, which is then forwarded to the base station. Function *estimate_ALPHA* calculates the value of ALPHA as explained in the next section.

3.2.4 Alpha Estimation

Composite threshold refers to a combined value of energy and signal strength calculated from the values in the information table of a mobile node. It is a measure of *quality* of the link with the corresponding neighbor. It is calculated using the equation:

*composite_threshold = (ALPHA * normalized_energylevel) + (1-ALPHA) * normalized_powerlevel*

where ALPHA is dynamically estimated. Battery power and signal power values stored in the information table of the nodes need to be normalized to the same scale before being used in the above given equation (Battery power is measured in joules and is usually a positive number in the range (0 - maximum power of the battery), while signal strength is measured in decibels is of the order of $10^{-}x$, where range of x depends on the wireless interface and other channel parameters. They are converted to ratios, denoted as a fraction of the maximum energy and maximum signal strength possible.

$$normalized\_energylevel = \frac{energy\ of\ neighboring\ node}{maximum\ energy\ possible\ for\ this\ node}$$
$$normalized\_powerlevel = \frac{signal\ strength\ for\ this\ link}{maximum\ signal\ strength\ possible\ for\ any\ link\ from\ this\ node}$$

The maximum possible values for node energy and signal strength of any link are pre-determined and remain constant for the entire duration of the simulation. The maximum possible value for the energy level would be the energy value of the batteries when fully charged. And the maximum possible value for signal strength for any link would be the signal power level calculated between two nodes that are very close to each other assuming ideal transmission conditions.

A static value for alpha provides a constant weight factor for the energy and power levels of nodes and links, irrespective of the current network topology. For example, at the start of the simulation, all nodes would have energy values close to the maximum. In this case, it would be better to have an ALPHA value of $\approx$ 0.2, thus giving more weight to the neighboring link power level. Similarly, when all the nodes are in close proximity to the main controller, the signal strength of the received packets would be close to maximum. An ALPHA value of $\approx$ 0.8 would be better, as the calculated composite threshold will be more biased towards energy values.

A simple method to dynamically estimate ALPHA value is adopted. The procedure is listed as an algorithm and is run once every *UPDATE_INTERVAL*. It is to be remembered that once every update interval, each node generates the update table and sends it to the main controller.

```
select best_node from update table where:
    best_node = node with maximum composite_threshold value;
select best_energylevel and best_powerlevel from information table where:
    best_energylevel = energylevel[best_node];
    best_powerlevel = powerlevel[best_node];
select local_maxenergy and local_maxpower from information table where:
    local_maxenergy = maximum energy value in the information table;
    local_maxpower = maximum power value in the information table;
if (local_maxenergy > best_energylevel)
     ALPHA = ALPHA + (1-best_energylevel/local_maxenergy)
if (local_maxpower > best_powerlevel)
     ALPHA = ALPHA - (1-best_powerlevel/local_maxpower)
```

It is to be noted that the energy and power level values of the node with maximum composite threshold need not necessarily be the maximum energy and power level values. Thus, if the maximum energy value in the information table is greater than the energy value of the node with the maximum composite threshold, ALPHA value is *increased* by the fraction of the difference between these two values. An increase in ALPHA value would result in higher weightage for the energy values in the computation of composite threshold. Similarly, if the maximum value for power level in the information table is greater than the power level of the node with maximum composite threshold, ALPHA value is *decreased* by the fraction of difference between these two values. Alpha estimation and the complete algorithm is explained using an example in Section 3.5.

3.3 Choice of AODV as the Routing Layer Protocol

The above discussed algorithm provides for node connectivity, but it does not provide for routing information between nodes. The forwarding of update_tables to main controller once every

UPDATE_INTERVAL relies on the underlying routing protocol for the transmission. The functionalities at the MAC layer required by an ad hoc network control layer, are very similar to that required by a wireless network. The IEEE 802.11 standard for wireless LAN's was chosen as the Medium Access Control (MAC) layer protocol, while the routing layer protocol had to be chosen from one of the several protocols designed specifically for ad hoc networks. The following considerations were made in choosing the routing protocol.

- The selected routing protocol should have a very minimal routing overhead. This suggest selection of on-demand routing protocols, since they have much lower overhead when compared to proactive protocols [34][27].
- Scalability is an important factor when making the choice of a suitable routing protocol. Increase in the number of nodes should not affect the performance of the system.
- Performance of the selected protocol should be well analyzed and compared with other standard protocols, as this gives us a base for comparing the performance of the proposed algorithm.
- Also, the routing protocol should have good performance in scenarios that involve constant topology change, and random node mobility, as these characterize the mobility of robots in USAR operations.

Based on the above mentioned factors the Ad hoc On-demand Distance Vector (AODV) was chosen as the routing protocol [17]. The Ad hoc On-Demand Distance Vector (AODV) algorithm enables dynamic, self-starting, multi-hop routing between participating mobile nodes wishing to establish and maintain an ad hoc network. It uses periodic broadcast of hello messages to maintain routes of established links. Energy information is appended to the hello messages, and the power level at which this packet is received by the neighboring nodes is estimated as an indicator for link strength. Thus the implementation of the proposed algorithm causes only a small incremental increase in routing overhead, and is studied in Section 4.3

Also from performance analysis of routing protocols for ad hoc networks in [26], it can be seen that AODV performs well in scenarios involving random node mobility and higher rates of movement. Also, at lower rates of mobility AODV and DSR perform better than the other protocols.

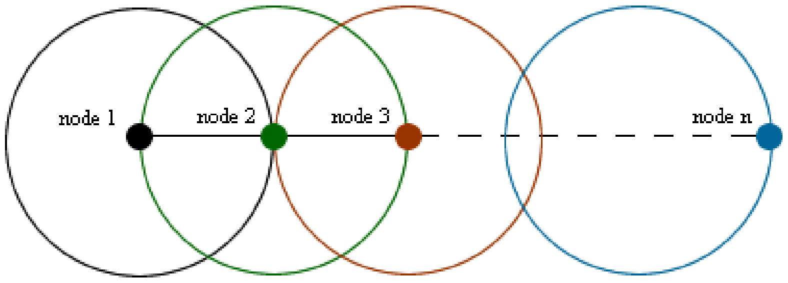

Figure 3. 5 Model Topology with Nodes in a Linear Chain

Also, the performance of AODV has been well analyzed and compared with other standard protocols [26, 27, 28, 29, 33, 35] for various scenarios and is also implemented in ns-2.26. AODV provides the right platform for building the control layer algorithms, without incurring any major routing overhead, and its implementation in ns gives a good base to compare and evaluate its performance against other standard routing protocols.

3.4 Variations Possible in the Proposed Algorithm

Certain design choices need a little bit more explanation.

- Instead of computing the composite threshold at every node, and sending this information to the main controller as an update table, each node could send its information table (consisting of neighbor id, energy level of the neighbor, and received signal power level), and the ALPHA value, and leave the computation to the main controller. This could help in reducing the computation time at the individual nodes, while it would increase the routing overhead. As can be seen from Table 3.1, information table has an extra *double* field when compared to update table (Table 3.2). Thus, depending upon the frequency of UPDATE INTERVAL, and the number of nodes in the network, there would be significant increase in the overhead due to transmitting the information table. If n is the number of nodes in the network, then for every $UPDATE\_INTERVAL$ there will be an extra $8 * n^2$ bytes of overhead, where 8 bytes is the default size for a $double$ variable.

Table 3.3 Information Table at Node 1 Along with Composite Threshold Values

Neighbor ID	Normalized Signal Strength	Normalized Energy	Composite Threshold
2	0.51	0.92	0.4692
3	0.57	0.81	0.4617
4	0.63	0.73	0.289
5	0.56	0.82	0.4592
BS	0.45	0.91	0.4095

- Instead of having the main controller validate the mobility of nodes by iterating through the update tables received from each node, the network could be flooded with update tables from each node, so that every node in the network has a copy of the update table of the other nodes. By doing this, the control of deciding on mobility is left to the individual nodes rather than the main controller. This would correspond to the *distributed* implementation of the proposed algorithm. Though this might seem plausible due to the fact, that each node could stop its mobility as and when it detects a weak link to the main controller, this approach has certain inherent drawbacks. Consider the example of a network with n nodes and a main controller, located in a linear chain (Figure 3. 5), such that each node is at the endpoint of the transmission range of its neighboring node. In such a topology, when using flooding techniques to broadcast information table to all nodes in the network, there are $(n-1)^2+n$ packets broadcasted, while if each node transmits its update table to the main controller, there is a total of $\sum_{a=1}^{n} a$ packets transmitted for every *UPDATE_INTERVAL*. The significant increase in number of packets exchanged and the consequent rise of congestion in the network when flooding, is one reason why each node sends its update table to the main controller and the main controller decides on mobility of the nodes. Moreover, keeping the computational complexity at the mobile nodes to a minimum, helps increase battery life.

- ALPHA values could be kept static, rather than having dynamically estimated values (Section 3.2.4). But with static ALPHA values the calculated composite threshold doesn't reflect the current network behavior. For example, lets assume a network of 5 nodes and a main controller, with static ALPHA value of 0.5 at each node. Let Table 3.3 be the values in the information table at node 1, along with their composite threshold values.

As can be seen from the information table for node 1, node 4 has the best link with this node (normalized signal strength of 0.63), while the neighbor node with the best energy is node 2 (normalized energy value of 0.92). The table also shows the composite threshold for the neighbors of node 1, calculated with $ALPHA = 0.5$. This results in node 2 having the maximum value for composite threshold, and would be chosen as the immediate parent [7] for this node. While, node 4 having a better signal strength, should have been made the parent neighbor for this node. An $ALPHA$ value of 0.1 would bias the calculation of composite threshold to the node with better power level, while an $ALPHA$ value of 0.9 would bias the calculation towards the neighbor node with better energy level. Thus, in order to balance the biasing factor, ALPHA values are dynamically estimated as explained in Section 3.2.4. Each node has its own $ALPHA$ value and is estimated based on the previous $ALPHA$ value, and the current data in the information table. The next chapter analyzes the effect of dynamic estimation vs. static ALPHA values on the performance of the algorithm.

3.5 Sample Model Based Explanation of the Algorithm

This section provides an illustration based explanation of the algorithm. Let us consider the example of a $700m*500m$ disaster site being explored with a main controller and 2 mobile robots. The main controller remains stationary during the entire course of search and rescue operations. All three nodes exchange hello packets containing the node energy value, once every *HELLO_INTERVAL*. The Information table is maintained at each node and stores this energy information for its neighbors, along with the power level at which the packet was received. Power level of the received packets are used as an estimator of signal strength. Let the transmission range for a node be 250m, and lets assume a link threshold of 0.32 (It is to be remembered that link threshold is the minimum value for composite threshold, below which a link is considered to be a weak link). Let the *HELLO_INTERVAL* be 1 second, *UPDATE_INTERVAL* be 1.2 seconds, and *MONITOR_INTERVAL* be 1.5 seconds. Also, *ALPHA* is set to 0.5 at all nodes, implying a constant weight factor between energy and power. Let all nodes move at a speed of 4 meters per second in a specific direction. Thus assuming that the nodes travel in straight direction, they would be out of the transmission

[7]Node having the maximum value for composite threshold among all the neighbors of a node is considered as the parent node.

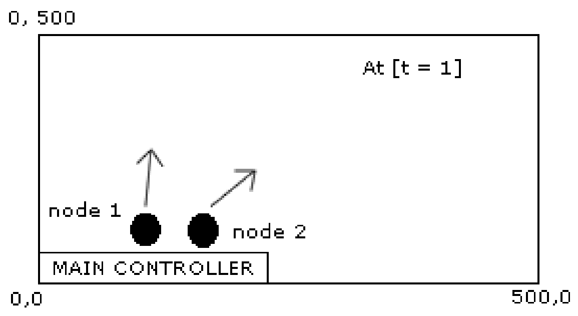

Figure 3. 6 Sample Model: Starting Topology

Table 3.4 Information Table at Node 1 and 2 at time t=1.0s

NODE 1			NODE 2		
Neighbor ID	Energy level	Signal Power	Neighbor ID	Energy level	Signal Power
2	0.95	0.9	1	0.95	0.87
BS	0.96	0.92	BS	0.93	0.9

range of the main controller in $\approx$ 50 seconds. Figure 3. 6 presents the network topology at $t = 1$. Nodes 1 and 2 are close to the main controller and their direction of movements are indicated. Hello messages are exchanged between the nodes, and data in the information table gets updated for each received hello message. Table 3.4 [8] shows the data stored in information table at time t=1. It is to be noted that energy and signal power values are normalized as explained in Section 3.2.4, while updating the contents of information table. At time t=1.2s the function call to send update table is evoked, that calculates composite threshold, estimates ALPHA value, and sends the update table to main controller. Using the equation for composite threshold given in Section 3.2.4, the update tables are generated at the individual nodes (Table 3.5).

[8]The values presented in these tables are not simulated, nor represent actual values taken from experiments, but were assigned to facilitate easy understanding of the algorithm

Table 3.5 Update Table at Nodes 1 and 2 at time t=1.2s

NODE 1		NODE 2	
Neighbor ID	Composite Threshold	Neighbor ID	Composite Threshold
2	0.925	1	0.91
BS	0.94	BS	0.915

To further illustrate the calculation of composite threshold, let us take the example of node 1. Node 2 is a neighbor of node 1 with energy level of 0.95 and signal power of 0.9. Hence the composite threshold for this neighbor of node 1 would be $(0.5 * 0.95) + (0.5 * 0.9)$ which is 0.925. After calculation of composite threshold, each node estimates its ALPHA value.

At node 1, the neighbor that has the maximum value for composite threshold is the main controller (from update table). Thus the best_energylevel and best_powerlevel (Section 3.2.4) correspond to the energy and signal power values of this node, i.e, 0.96 and 0.92 respectively. Also, from the information table, it can be seen that the local_maxenergy and local_maxpower correspond to the values of main controller. This implies that the current value of ALPHA is properly biased between energy and signal strength, and thus the ALPHA value for this node remains the same.

Again among the neighbors of node 2, the main controller has the maximum value for composite threshold. Variables best_energylevel and best_powerlevel correspond to values 0.93 and 0.9 respectively. But the values for local_maxenergy and local_maxpower correspond to 0.95 and 0.9 respectively (Table 3.4). Despite having a greater energy value, node 1 doesn't have the maximum composite threshold value. Thus the alpha value is biased to give more weight to energy value in the calculation of composite threshold. Since max(normalized energy) $>$ normalized energy of node with max(composite threshold),

$$
\begin{aligned}
ALPHA &= ALPHA + (1 - best\_energylevel/local\_maxenergy) \\
&= 0.5 + (1 - 0.93/0.95) = 0.521
\end{aligned}
$$

ALPHA value for node 1 remains at 0.5, while node 2 now has an ALPHA value of 0.521, increased weight for energy value of neighbors.

After ALPHA estimation, each node sends its update table to the main controller. On receiving these tables from the individual nodes, the main controller updates its data structure to reflect the current network topology. Thus at $t = 1.2s$, the main controller has a data structure similar to Table 3.6 At every *MONITOR_INTERVAL* (=1.5 in this case) the main controller runs its local algorithm to check for links from all nodes to itself. From Table 3.6, it can be seen that nodes 1 and 2, both have a direct connection with the main controller, and is above the link threshold (=0.32). So both nodes can remain mobile, and the mobility_flag for these nodes is set to true.

Table 3.6 Data Structure at Main Controller at time t=1.2s

Node ID	Neighbor ID	Composite Threshold	Mobile
1	2	0.925	Yes
	BS	0.94	
2	1	0.91	Yes
	BS	0.915	

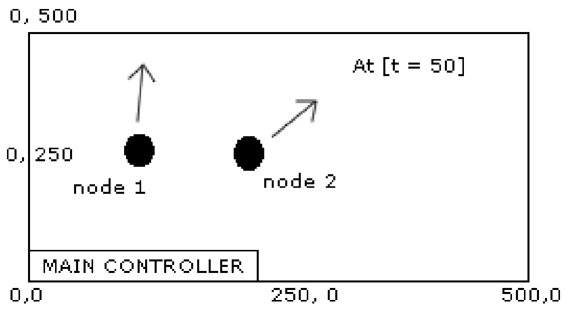

Figure 3. 7 Sample Model: Network Topology at time t=50.0s

Figure 3.5 represents the network topology at $timet = 50.0s$. Table 3.7 shows the information table at nodes 1 and 2, updated based on the hello messages received from their neighbors at $timet = 50.0s$.

Again at t=50.4s, the function call to update table is evoked. Since ALPHA values are dynamically estimated every UPDATE INTERVAL, using the value of 0.5 and 0.521 for nodes 1 and 2, estimated at t=1.2s would not be appropriate. Thus ALPHA is assumed to be 0.2 at t = 50.0s. The generated update table is shown in Table 3.8. ALPHA values are estimated in the same way as in the previous case and is not shown here. At t=50.4s, the nodes forward their update tables to the main controller, where the local data structure is updated based on the data received from the nodes in the network. The updated data structure at the main controller is shown in Table 3.9

Table 3.7 Information Table at Node 1 and 2 at time t=50.0s

NODE 1			NODE 2		
Neighbor ID	Energy level	Signal Power	Neighbor ID	Energy level	Signal Power
2	0.75	0.85	1	0.72	0.8
BS	0.8	0.2	BS	0.8	0.1

Table 3.8 Update Table at Nodes 1 and 2 at time t=50.4s

NODE 1		NODE 2	
Neighbor ID	Composite Threshold	Neighbor ID	Composite Threshold
2	0.83	1	0.784
BS	0.32	BS	0.24

Table 3.9 Data Structure at Main Controller at time t=50.4s

Node ID	Neighbor ID	Composite Threshold	Mobile
1	2	0.83	Yes
	BS	0.32	
2	1	0.784	Yes
	BS	0.24	

The main controller executes its local algorithm at t=51s (*MONITOR_INTERVAL* = 1.5s). As can be seen from Table 3.9 node 1 has a direction with the main controller, but the composite threshold value is at the link threshold. So, the algorithm loops through the update tables of other neighbors of this node, which have a composite threshold value greater than link threshold, for a connection to the main controller. The only other neighbor for node 1 is node 2, and its composite threshold value is greater than link threshold. Hence the algorithm checks the neighbors list of node 2 for a strong link to the main controller. But node 2 has an even weaker connection to main controller (composite threshold = 0.24), and hence the algorithm stops the mobility of node 1 since its only connection to the main controller is at threshold. The mobility flag of node 1 is set to false, and a message is sent to the node with the mobility flag.

Similarly, the algorithm looks for a connection from node 2 to the main controller. The direct link from node 2 to BS is below the link threshold (=0.24), and the algorithm checks for a connection to main controller through other neighbors of this node which have threshold values greater than link threshold. Neighbor node 1 has a connection with the main controller, which is at link threshold. But since its mobility has already been stopped, and is not in danger of breaking away from the main controller, the algorithm sets node 1 as the parent node of 2, and node 2 continues to have its mobility flag set to true.

Figure 3. 8 presents the network topology at t=100s. It is to be remembered that node 1 has its mobility flag set to false, and node 2 has its mobility based on node 1, i.e, node 2 has a link to the main controller through node 1. ALPHA value is assumed to be 0.3 at t=100s.

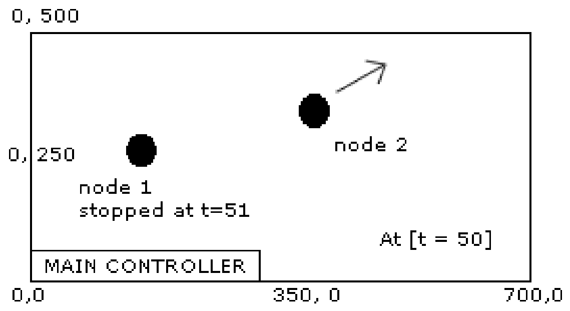

Figure 3. 8 Sample Model: Network Topology at time t=100.0s

Table 3.10 Information Table at Node 1 and 2 at time t=100s

NODE 1			NODE 2		
Neighbor ID	Energy level	Signal Power	Neighbor ID	Energy level	Signal Power
2	0.5	0.4	1	0.3	0.328
BS	0.6	0.2			

Based on the hello messages exchanged between the nodes at t=100s, the information table of each node gets updated and is shown in Table 3.10. It is to be noted that there is no entry for main controller in the table of node 2, and this is due to the fact that node 2 has moved well beyond the transmission range of main controller and the hello messages broadcasted by the main controller are not received at node 2. At t=100.4s, the update table function is evoked by every node, which generates the update table (Table 3.11) for that node, estimates ALPHA value and sends out the update table to main controller. The contents of the data structure at the main controller get updated on receiving these tables form the mobile nodes and is shown in Table 3.12.

Node 1 has its mobility flag set to false and the algorithm at the main controller is not able to find any better link to the main controller (this occurs if a node with a strong link to the main controller moves within the transmission range of node 1). But node 2, now has only one neighbor, node 1,

Table 3.11 Update Table at Nodes 1 and 2 at time t=100.8s

NODE 1		NODE 2	
Neighbor ID	Composite Threshold	Neighbor ID	Composite Threshold
2	0.28	1	0.32
BS	0.32		

Table 3.12 Data Structure at Main Controller at time t=100.8s

Node ID	Neighbor ID	Composite Threshold	Mobile
1	2	0.28	No
	BS	0.32	
2	1	0.32	Yes

and this link has a composite threshold value equal to link threshold. Node 2 can still communicate with the main controller through node 1, but both the links are at threshold limits.

The algorithm at the main controller iterates through the data collected from update tables and checks for mobility of the nodes. A timer is attached to every node to keep track of the duration for which it has been stopped. Based on this time value, the main controller can issue *callback* functions to the nodes requesting them to move towards the base.

The example topology presented in this section provides the basic idea of the algorithm detailed in the previous sections. The next chapter discusses the implementation parameters, actual values used for simulations, changes to the base algorithm done during implementation, simulation scenario and finally analyzes the results of simulation and efficiency of the proposed algorithm.

CHAPTER 4

IMPLEMENTATION AND EXPERIMENTAL RESULTS

This chapter describes the implementation of the proposed control layer algorithm in *ns*-2.26, explains the simulation setup, and analyzes the performance of proposed algorithm based on link duration, overhead and throughput.

4.1 Implementation in ns-2.26

ns (network simulator) is a comprehensive tool to simulate wired and wireless networks. Wireless extensions from the MONARCH project at Carnegie Mellon [26] provides for node mobility, realistic physical layer including a radio propagation model supporting propagation delay, capture effects and carrier sense, radio network interfaces with properties such as transmission power and antenna gain and implements IEEE 802.11 Medium Access Control (MAC) protocol using the Distributed Coordination Function (DCF).

4.1.1 Physical Layer

The wireless interface works like the 914 MHz Lucent WaveLAN Direct-Sequence Spread-Spectrum (DSSS) radio interface [42]. WaveLAN is modelled as a shared-media radio with a nominal bit rate of 2 Mb/s, and a nominal radio range of 250m. Each mobile node uses an omni-directional antenna with unity gain. The antenna is positioned to be at the center of the mobile node and 1.5 meters above the ground (X_ = 0, Y_ = 0, Z_ = 1.5, Gr = 1.0, Gt = 1.0, where Gr and Gt are the gain of the transmitter and receiver antennas respectively).

The SharedMedia interface is initialized with the following parameters to make it work like the 914MHz Lucent WaveLAN DSSS radio interface.

- Carrier Sense Threshold (CSThresh_)=$1.559e^{-11}$ Watts or -78dBm

- Receiving Threshold (RXThresh_)= $3.652e^{-10}$ or -64dBm
- Bandwidth = 2 Mbps
- Frequency = 914 MHz

The signal propagation model combines both a free-space propagation model and a two-ray ground reflection model. The free-space model is used when the transmitter is within the *cross-over distance* of the receiver, and the two-ray ground model is used otherwise. The free-space model uses *Friis equation* (Equation 4.1) to estimate the received signal power, where the signal attenuates as $1/d^2$.

$$P_R = \frac{\lambda \, G_T \, G_R \, P_T}{(4\pi)^2 \, d^2} \tag{4.1}$$

where, λ is the wavelength, P_R is the received signal power in Watts (or dBm), G_T is gain of the transmitter's antenna, G_R is the gain of the receiver's antenna, P_T is the transmitted signal power in Watts (or dBm) and d is the distance between the receiver and transmitter antennas measured in meters. In the two-ray ground reflection model [43], the received signal power is inversely proportional to d^4 or attenuates as $1/d^4$, and is estimated using equation 4.2.

$$P_R = \frac{G_T \, G_R \, P_T \, ({H_T}^2 \, {H_R}^2)}{d^4 \, L} \tag{4.2}$$

where H_T and H_R are the heights of the transmitter and receiver antennas respectively. The cross-over distance is calculated using equation 4.3.

$$Crossover \; distance = \frac{4 \, \pi \, H_T \, H_R}{\lambda} \tag{4.3}$$

Each mobile node has a velocity and position information associated to it. The position of a node is calculated as a function of time and is used by the propagation model to estimate the received signal power. When a packet is received at a mobile node, its received power is estimated by the physical layer interface. The packet is discarded if the received power is below the carrier sense threshold, and is marked as error before being passed to the MAC layer, if the received signal power is between the carrier sense threshold and receive threshold. Otherwise, the packet is handed to the MAC layer as a good packet.

4.1.2 Data Link Layer

The link layer of ns-2.26 implements the complete IEEE 802.11 standard [8] Medium Access Control (MAC) protocol to model the contention of nodes for the wireless medium. The main functions/features of this layer are listed below:

- The IEEE 802.11 MAC standard implements Distributed Coordination Function (DCF), to use both virtual and physical sensing mechanisms to reduce the probability of collisions due to the hidden terminal problem.
- The transmission of any unicast packet is preceded by exchange of Request-to-Send/Clear-to-Send (RTS/CTS) that reserves the wireless channel for the transmission of a data packet.
- The sender receives an acknowledgement from the receiver, for every correctly received packet, until which time it retransmits the packet at specific interval.
- Broadcast packets are not preceded by RTS/CTS exchange, but are sent only when the virtual and physical carrier sense indicate that the medium is clear for communication. Broadcast is unreliable in 802.11 because no acknowledgement is sent.
- It implements a binary back-off mechanism to reduce the likelihood of collisions.

A detailed study of the IEEE 802.11 standard and its implementation in ns-2.26, can be found in [8] and [39] respectively.

4.1.3 Control Layer

Extensions were added to the simulator to implement the control layer algorithm. Separate control layer modules were developed for the main controller and the mobile nodes. ALPHA estimation is kept local, and each node maintains a local copy of ALPHA calculated once every UPDATE INTERVAL, based on the data in its information table. The control layer algorithm (Figure 3. 2) relies on the underlying MAC layer for the exchange of broadcast hello packets and the estimation of the signal power, and on the routing layer for the transmission of update tables to the main controller. We assume a HELLO INTERVAL and UPDATE INTERVAL of 1 second, and a MONITOR INTERVAL of 1.5 seconds. A time variable from a standard random distribution is added to both

HELLO INTERVAL and UPDATE INTERVAL to reduce the contention for the medium. Hello messages are exchanged once every second, and the mobile nodes generate the update table and forward it to the base station at the same interval. The base station, runs its algorithm to check for safe mobility of the nodes once every MONITOR INTERVAL, i.e. 1.5 seconds.

The receiving threshold (RXThresh_) for the shared media radio interface is $3.652e^{-10}$, which implies that packets received with signal power less than RXThresh_ are discarded due to low power level or are marked as packets in error. Based on this value of RXThresh_ we assign a *link threshold* value of $3.652e^{-9}$, i.e, packets received through a link at a signal power less than link threshold are considered weak links. Such links are not counted for, when the control layer algorithm checks for connection to the main controller. The maximum signal power at which a packet can be received would be when two nodes are at a very close distance, and can be calculated using Friis equation (Equation 4.1). With a transmission power of 0.2818 Watts (P_T = 0.2818 Watts for a 100m transmission range assuming AT&T's WaveLAN PCMCIA card), distance between nodes as 0.1m, unit transmitter and receiver gain, and using a 914Mhz WaveLAN PCMCIA card, the maximum received signal power could be 0.0585 Watts (substituting values in Equation 4.1). Thus for a link to be considered a strong link, packets need to be received at a signal power level in the range from 0.0585 Watts to $3.652e^{-9}$ Watts. This maximum and minimum value for signal power is used in estimating the normalized signal power.

The composite threshold is estimated based on the normalized signal power, and normalized battery power. Initially all nodes have a battery power of 600 Joules. Normalized battery power at time τ is calculated as a ratio of the $\frac{remaining\ battery\ power\ of\ neighbor\ node\ at\ time\ \tau}{maximum\ battery\ power\ (600J)}$. To estimate the normalized signal power, the maximum and minimum values for signal power to establish stable links are divided (0.03652W and $0.3652e^{-10}$W) into 20 divisions on a log scale. Each division has a value of 0.05, thus based on the received signal power, the normalized signal power can range from 0 to 1. It is to be noted that the derived value of 0.0585W for maximum possible signal power is not used, and instead the maximum possible signal power OF 0.03652W is utilized. Normalized signal power is assigned a value of 1, if the received signal power is above 0.03652W.

4.1.4 Routing Layer

The ad hoc routing protocols DSR, DSDV, AODV and TORA have been implemented in ns, with AODV being the routing protocol chosen for this work (see Section 3.2.4). As elaborated in Section 3.3, we choose AODV as the routing protocol. The implementation of AODV in ns follows the standard described in [17]. However the implementation differs slightly from the specifications in the method of detecting link breakages. Instead of using *hello messages* to detect link layer breakages, the authors in [26] analyze the effect of using physical layer methods. This small change modifies the base behavior of the protocol because it eliminates the overhead of the hello messages. Though this saves the overhead due to periodic hello messages, it increases the convergence time and modifies the basic behavior of the protocol. However, since the proposed control layer algorithm makes use of hello messages to propagate energy and power information, the original implementation of AODV is utilized that uses periodic hello messages to detect link layer breakages.

4.2 Simulation Scenario

The evaluation of the proposed algorithm is based on the simulation of 9 wireless nodes (8 mobile nodes or robots and 1 stationary node or main controller) forming an ad hoc network, moving over an area of 600m x 600m flat space for 600 seconds of simulated time. These are very realistic number for USAR applications. Each node has 600 J of energy at the start of simulation with a drain of 0.5 W for every packet received and a drain of 0.4 W for every packet sent. The physical characteristics of the mobile nodes network interface is modelled to approximate the Lucent WaveLAN DSSS radio interface. The physical channel replicates the two-ray ground propagation model. Throughput, link stability, and overhead are used as performance metrics. Intuitively, the control layer algorithm should provide for stable links throughout the course of the simulation. By ensuring node connectivity, the control layer algorithm also ensures that there is a constant communication link between all nodes in the network, i.e., a tree structure of the network is always preserved. Link stability is evaluated by establishing UDP/TCP flows between all nodes and continuously monitoring the status of these flows. The correctness of the algorithm is tested for 6 different mobility models with the same communication model and physical parameters.

4.2.1 Mobility Model

It is imperative to evaluate the proposed algorithm utilizing different mobility models as the performance of an ad hoc network can vary significantly with different mobility patterns [44]. A survey of the various models is presented in [44]. The selected mobility model needs to represent the movement behavior of robots in search and rescue operations. The mobility of autonomous robots is controlled by its sensor inputs and its programmed response behavior. This essentially means that the robots move around in a random fashion, not following a pre-defined pattern. The random waypoint mobility model [45] replicates this behavior: nodes move towards a particular destination at a selected speed, pause for *pause time*, select a random destination and a new speed and start moving towards this new destination. This is repeated throughout the course of the simulation. Random waypoint model (2 cluster based models and a generic model), random walk model, a group mobility model (column based mobility where all nodes move towards the same destination at the same speed) and a special case of the random waypoint mobility model are used to evaluate the performance of the proposed algorithm. In the modified version of random waypoint model, nodes select a particular destination and move towards this destination at a set speed and do not move any further after reaching the destination, while in the base model after a brief pause time they would pick another destination and start moving towards it. In the results section (Section 4.3), the mobility model of the nodes is first introduced, followed by an analysis of the throughput, link stability and overhead graphs for each model.

4.2.2 Communication Model

All the mobile nodes are connected to the main controller through FTP flows that exist for the entire duration of simulation (Packet size is 512 bytes). Similarly FTP connections are established from the main controller to the individual nodes with the same parameters. Monitoring these flows gives information on the stability of these links and the duration for which the flow has been active. Similarly, to study the effect of the proposed algorithm on the overall link stability of the system, very low bit rate UDP connection are created between all the mobile nodes of the network (0.8Kbits or 100 bytes per second). This forms a completely connected graph structure of *n* nodes, where the total number of links/flows is equal to $n * (n - 1)$, which is a total of 72 flows in the simulations

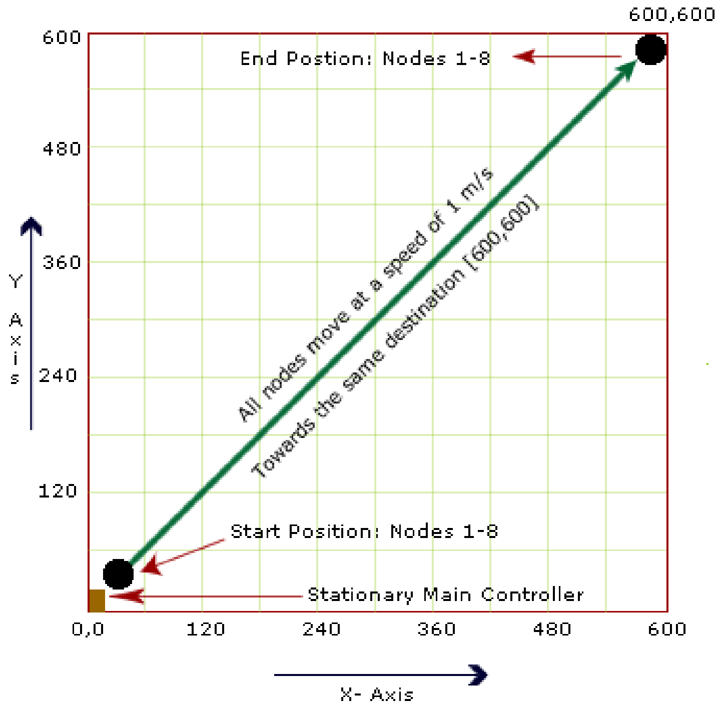
600,600
End Postion: Nodes 1-8
All nodes move at a speed of 1 m/s
Towards the same destination [600,600]
Start Position: Nodes 1-8
Stationary Main Controller
Y Axis
X- Axis
0,0
120
240
360
480
600

Figure 4. 1 Mobility Model 1 (Straight Line Movement Along the Diagonal)

(8 mobile nodes and a main controller). The traffic model used is a representative of the kind of communication desired between robots operating in disaster sites. All the robots constantly communicate with the main controller (FTP connections), and also keep sending minimal information to neighbor nodes (CBR connections).

4.3 Results

This section discusses the simulation results and analyzes the performance of the proposed algorithm. 6 mobility models have been used and a brief description of each of the models precede the performance analysis for that model.

4.3.1 Simulation Model 1: Group Mobility

The first model implements a group mobility pattern (column mobility model)[44]. The main controller is stationed at the location (0,0) and the 8 mobile nodes start at (0,0) and move along the diagonal towards (600,600) at a rate of 1 m/s (Figure 4. 1). The ideal transmission range of the

Lucent WaveLAN DSSS radio interface is 250m. This implies that nodes moving at 1 m/s would break its communication with the main controller at around 250 seconds. This would be the case when the network uses *any* of the existing WLANs. But when the network is operated with the AODV based control layer algorithm, there should be a constant communication link from all the nodes to the main controller. Simulations were run for this mobility model with the existing AODV (which is referred to as *base AODV*) and AODV with the control layer algorithm. The results of simulation are presented below.

The most important of all plots is the link stability plot. As discussed in the previous section we have a total of 72 flows (n completely connected nodes, n(n-1) links): 16 FTP flows from the mobile nodes to the main controller and vice versa and 56 CBR flows, at a very low data rate, interconnecting all the 8 mobile nodes. Flow ID 1-56 denotes the UDP/CBR connections between the mobile nodes, while flow ID's 57-64 denotes connections from the main controller to the mobile nodes and flow ID's 65-72 denotes connection from mobile nodes to main controller. The communication between mobile nodes (CBR flows 1-56) would always be active as the nodes are moving together. However, at around 250 seconds in the simulation all the nodes would loose their communication with the main controller, indicating a break in the flows 57-72. But when using AODV with the control layer algorithm, this is not the case, and the link stability plot in Figure 4. 2 indicates that a constant communication link is established for all flows throughout the duration of the simulation.

This can be verified from the throughput plots in Figures 4. 3 and 4. 4, the former represents the overall network throughput for all flows, while the latter corresponds to the throughput at the main controller. From the latter plot it is evident that throughput of base AODV drops to zero at 250 seconds, when the connections with the mobile nodes are terminated. However AODV with control layer ensures a steady throughput throughout the duration of the simulation. From the overall throughput plot in Figure 4. 3, it can be seen that the network throughput does not drop to zero for base AODV and this is due to the CBR flows between the mobile nodes. As in the previous case, AODV with control layer algorithm provides a steady throughput throughout the duration of simulation. It is to be noted that for the first 250 seconds of simulation when all flows are active, the throughput achieved by both the protocols are comparable, indicating that the network throughput is not affected by the overhead due to the control layer algorithm.

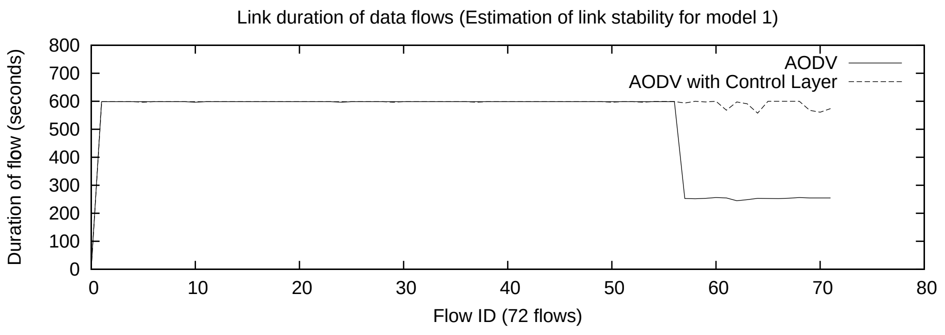

Figure 4. 2 Comparative Link Stability Analysis for Mobility Model 1

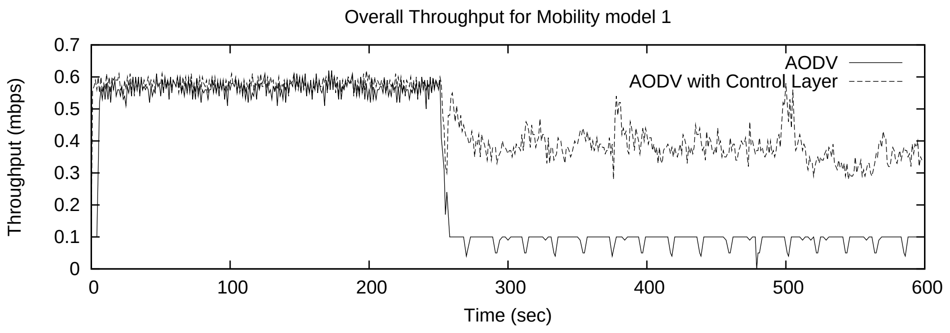

Figure 4. 3 Throughput of All Flows for Model 1

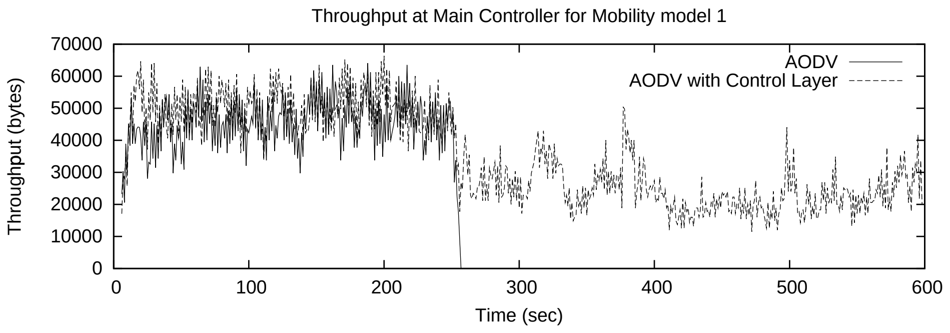

Figure 4. 4 Throughput at Main Controller for Model 1

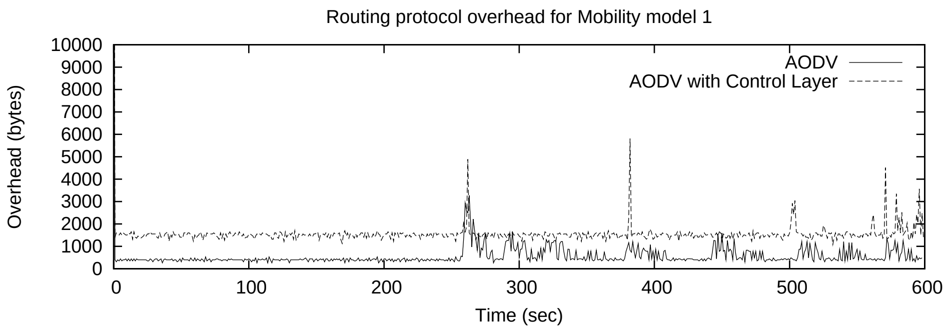

Figure 4. 5 Network Overhead (Routing and Hello Packets) for Model 1

Figure 4. 5 plots the overall routing overhead (which includes the hello packets, update packets sent to the main controller, and AODV control packets) of the two compared protocols. AODV with control layer algorithm introduces a mean increase in overhead of 1200 bytes per second, at the rate of 4.8Kbps. Obviously this increase in overhead is very minimal when compared with the network bandwidth of 2Mbps and the performance improvements achieved by using the control layer algorithm.

From the simulation results and plots for model 1, it is clear that AODV with control layer algorithm achieves constant node connectivity and steady throughput without incurring any major increase in routing overhead.

4.3.2 Simulation Model 2: Cluster Based Random Waypoint Model

In this model a cluster based movement of the nodes is simulated. The network topology is as shown in Figure 4. 6. Nodes 1, 2, 3 and 4 form a cluster and move towards the x-axis at y=600. Similarly nodes 5, 6, 7 and 8 form a cluster and move towards the y axis at x=600. Other simulation parameters are the same as in the previous model. Intuitively, when using the base AODV algorithm, the nodes would loose their connectivity with the main controller at around 250 seconds. However unlike in the previous model where the flows among mobile nodes was always active, there would not be any connection between the nodes of the two clusters after t=250s. But the nodes within the same cluster could communicate with each other throughout the course of the simulation. Thus the link stability plot should indicate alternating stable durations of 600s and 250s for flows 1 to 56 and the duration of flows 57 to 72 would be around 250s. The link stability plot for model 2 is presented in Figure 4. 7. The results are as expected and even in this model, AODV with control layer algorithm provides stable links for all the flows. It is seen that the implementation of control layer algorithm not only improves the stability of the links with the main controller, but also the links between the mobile nodes.

The overall network throughput, throughput at main controller and overhead for model 2 are shown in Figures 4. 8, 4. 9 and 4. 10 respectively. As in the previous model, AODV with the control layer algorithm ensures constant throughput throughout the course of the simulation, while the throughput of the base AODV algorithm drops at 250s, when the mobile nodes break away from

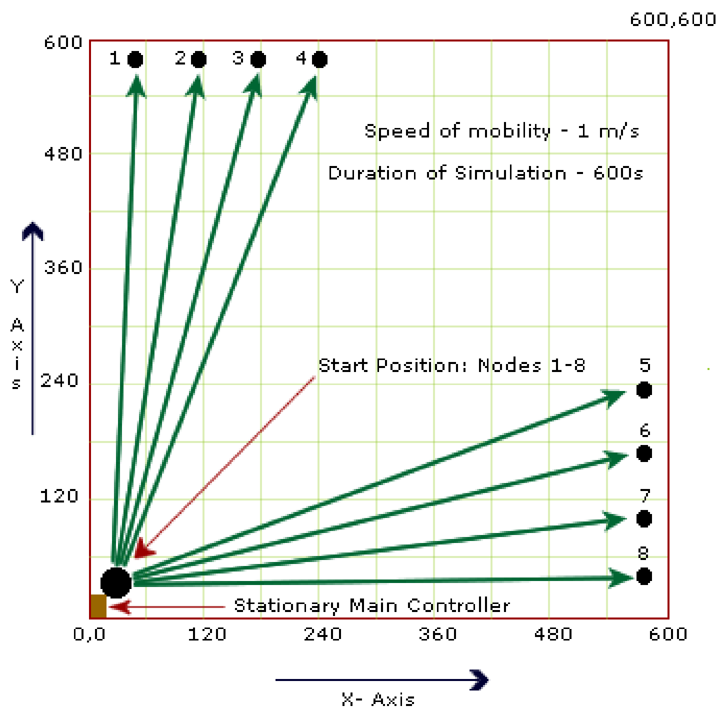
600,600
600
480
360
240
120
Y Axis
1
2
3
4
Speed of mobility - 1 m/s
Duration of Simulation - 600s
Start Position: Nodes 1-8
5
6
7
8
Stationary Main Controller
0,0
120
240
360
480
600
X- Axis

Figure 4. 6 Mobility Model 2 (2 Clusters Moving Towards Preset Destination)

the main controller. Similarly, the increase in overhead due to control layer packets is minimal and is outweighed by the achieved performance improvement.

4.3.3 Simulation Model 3: Random Waypoint Model (Independent Direction)

In this model, the main controller is stationed at the center of the disaster site (300,300) and all the mobile nodes start from this location and move towards the perimeter of the simulation area. Each node moves in a different direction, at an angle of 45 degrees from its neighbors. Intuitively, all nodes would loose their connection with the main controller (flows 57-72) at around 250 seconds (maximum transmission range is 250m and nodes are moving at 1m/s), however all the inter-nodal connections (flows 1-56) would break at around 350 seconds (right angle triangles with 250m sides would have an hypotenuse $\approx$ 350 meters). The link stability plot in Figure 4. 12 shows this behavior. When using the base AODV, the communication link with the main controller is broken at 250s (flows 57-72), while the inter-nodal communication (flows 1-56) breaks at 340s. However AODV with control layer algorithm ensures constant link stability throughout the duration of simulation.

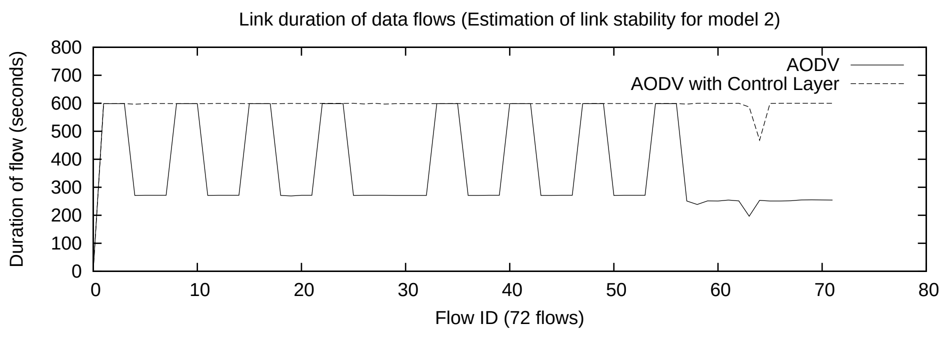

Figure 4. 7 Comparative Link Stability Analysis for Mobility Model 2

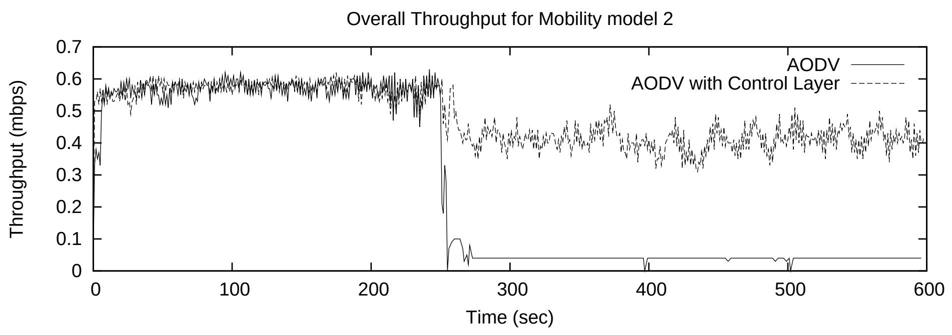

Figure 4. 8 Throughput of All Flows for Model 2

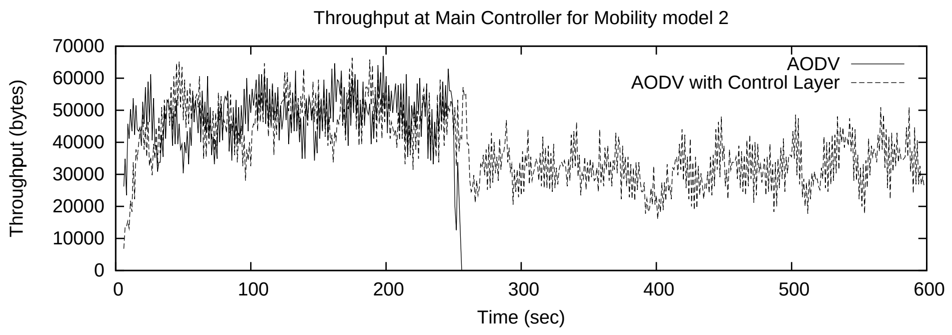

Figure 4. 9 Throughput at Main Controller for Model 2

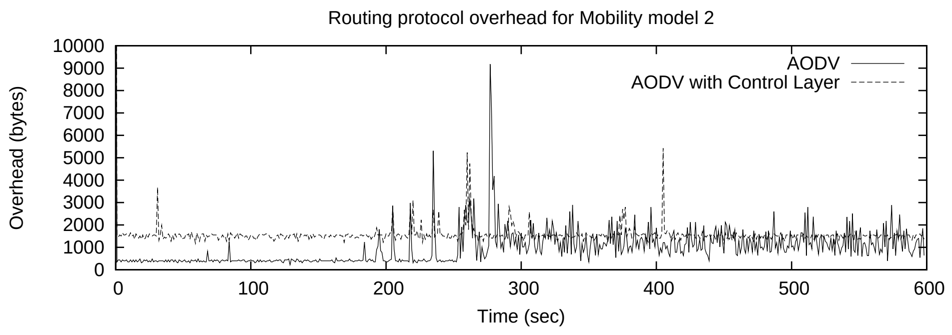

Figure 4. 10 Network Overhead (Routing and Hello Packets) for Model 2

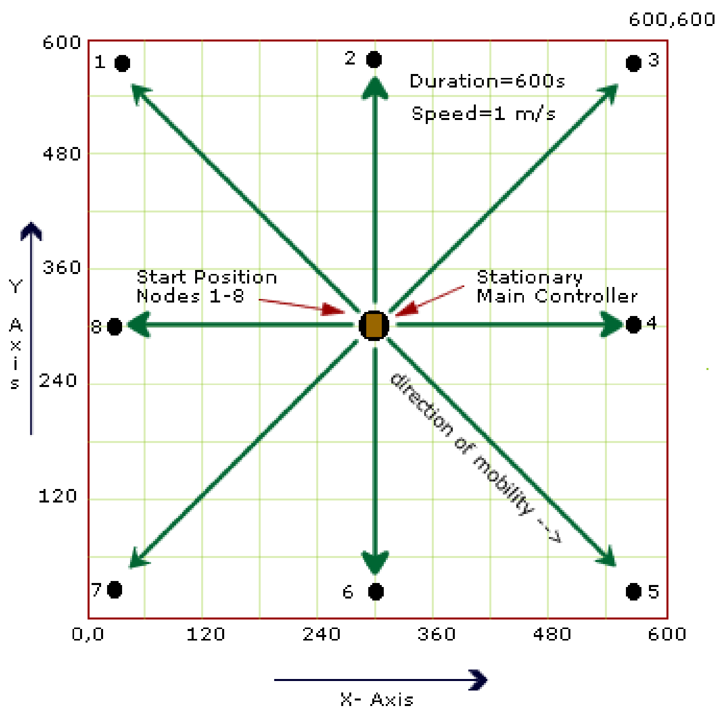

Figure 4. 11 Mobility Model 3 (Nodes Moving at an Inter-nodal Angle of 45°)

The throughput plots shown in Figures 4. 13 and 4. 14 represent the same behavior. The throughput, when using base AODV, drops to 0.1 Mbps after 250 seconds of simulation and to zero after 350 seconds. The initial drop is due to the breakage of links between the mobile nodes and the main controller while the second drop is due to the breakage of links between the mobile nodes. However, AODV with control layer algorithm ensures a steady throughput for the entire course of simulation. Figure 4. 15 compares the network overhead due to AODV, hello and control layer packets. Clearly there is a mean increase of 1200 bytes in overhead, but this increase is insignificant comparing the available bandwidth and the achieved performance improvements.

4.3.4 Simulation Model 4: Random Walk Model

In this model the main controller is stationed at (0,0) and all the other mobile nodes start from this location (Figure 4. 16). The mobile nodes select a random destination (>250m) and move towards this destination at a rate of 1m/s. No two nodes move towards the same destination. The random mobility of the nodes makes it tough to predict the link stability and throughput of the

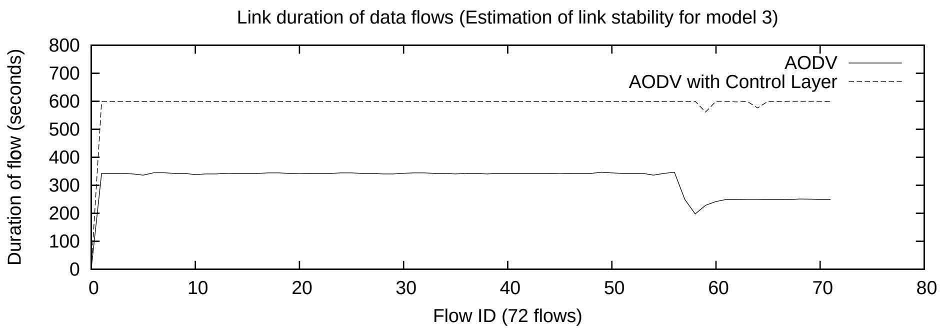

Figure 4. 12 Comparative Link Stability Analysis for Mobility Model 3

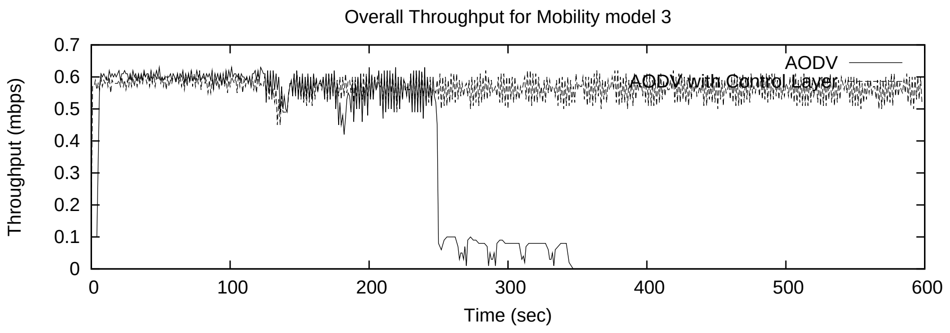

Figure 4. 13 Throughput of All Flows for Model 3

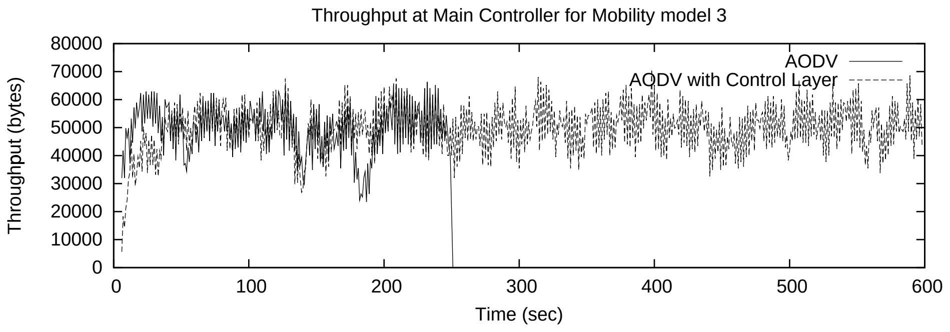

Figure 4. 14 Throughput at Main Controller for Model 3

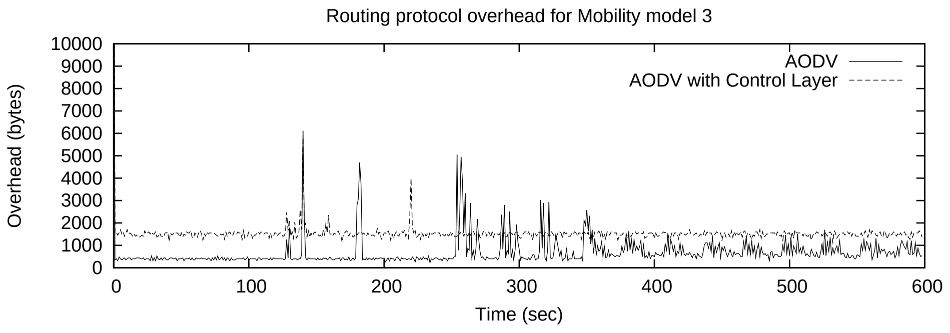

Figure 4. 15 Network Overhead (Routing and Hello Packets) for Model 3

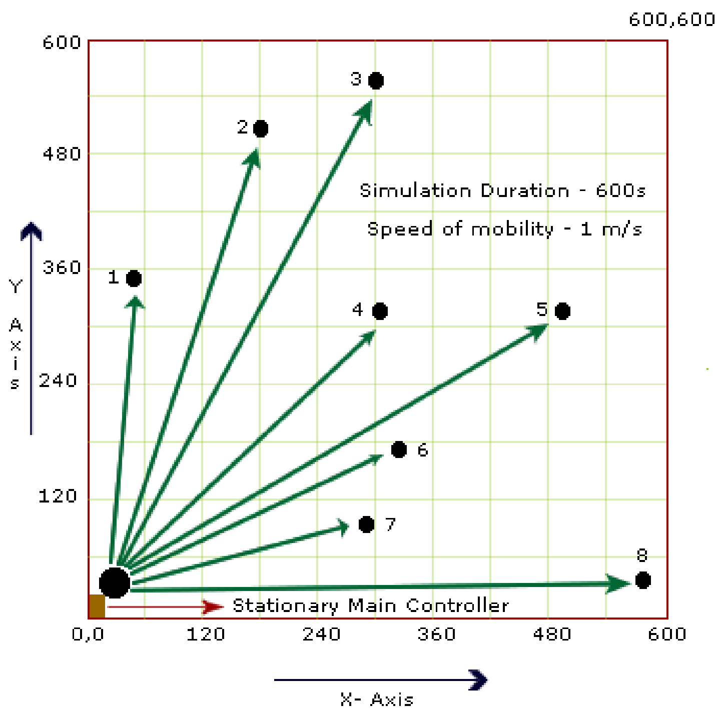
600,600
600
480
360
240
120
0,0
120
240
360
480
600
Y Axis
X- Axis
1
2
3
4
5
6
7
8
Simulation Duration - 600s
Speed of mobility - 1 m/s
Stationary Main Controller

Figure 4. 16 Mobility Model 4 (Random Walk Model)

network. However, it is imperative that the results for AODV with control layer algorithm should ensure node connectivity throughout the duration of simulation. A comparative link stability analysis of the network is presented in Figure 4. 18. As can be seen from the plot, when using base AODV all nodes loose their communication link with base station at 250s, while AODV with control layer algorithm ensures stable links throughout the simulation. The inter-nodal communication remains stable for the entire simulation time, when using AODV and AODV with control layer algorithm. But, during the time where complete connectivity exists in both algorithms (0-250 seconds), the network throughput graph in Figure 4. 19 shows a marginal difference in the throughput achieved by the two different protocols. AODV with control layer has a lower throughput (0.1Mbps lesser than base AODV), and this could be attributed to the overhead incurred by sending the control layer packets. A similar difference in throughput is observed in Figure 4. 20 and also for the other mobility models. Figure 4. 21 plots the routing protocol overhead of the two methods and like in previous models there is a mean increase of 1200 bytes in overhead when using AODV with control layer algorithm.

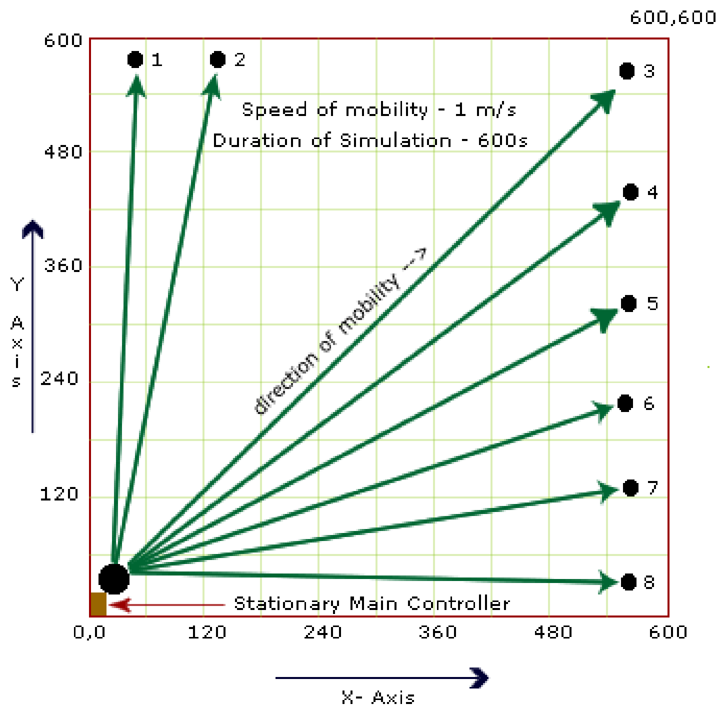
600,600
600
480
360
240
120
Y Axis
1
2
3
4
5
6
7
8
Speed of mobility - 1 m/s
Duration of Simulation - 600s
direction of mobility -->
Stationary Main Controller
0,0
120
240
360
480
600
X- Axis

Figure 4. 17 Mobility Model 5 (Nodes Moving in 2 Different Clusters)

4.3.5 Simulation Model 5: Random Waypoint Model (Cluster Based Mobility)

The model presented in Figure 4. 17 is similar to mobility model 2, but for the fact that there are 6 nodes in one cluster and 2 in the other. However, in mobility model 2, there were 4 nodes in each cluster, and the communication between the nodes of the two clusters was dependent on their connection with the main controller. When the nodes in each cluster lost their link with the main controller, it resulted in a link breakage between the two clusters. But in the model shown in Figure 4. 17, nodes in the two clusters can communicate even after their communication with the main controller is terminated. For example, nodes 2 and 3 can maintain a communication link for a certain duration even after their communication with the main controller is terminated. If there is a connection between one node in a cluster and another node in the other cluster, then all nodes in each cluster can route packets through these nodes (nodes 2 and 3 in this case). But, when these 2 nodes in different clusters loose their connection, the two clusters become completely independent. Link stability plot shown in Figure 4. 22 confirms this kind of a behavior. As in earlier models, the mobile nodes break away from the main controller (flows 57-72) after 250 seconds into simulation

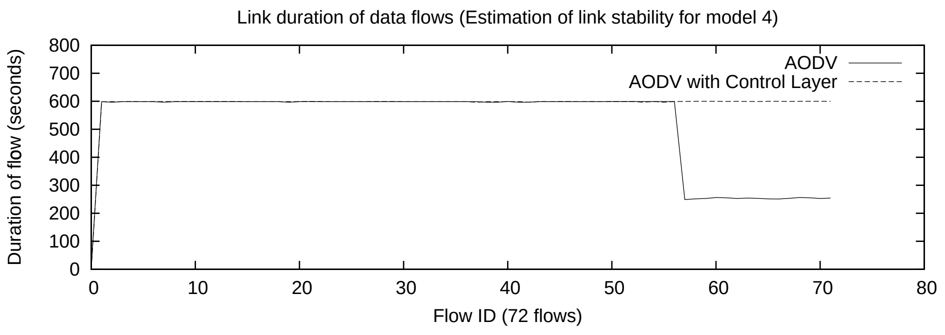

Figure 4. 18 Comparative Link Stability Analysis for Mobility Model 4

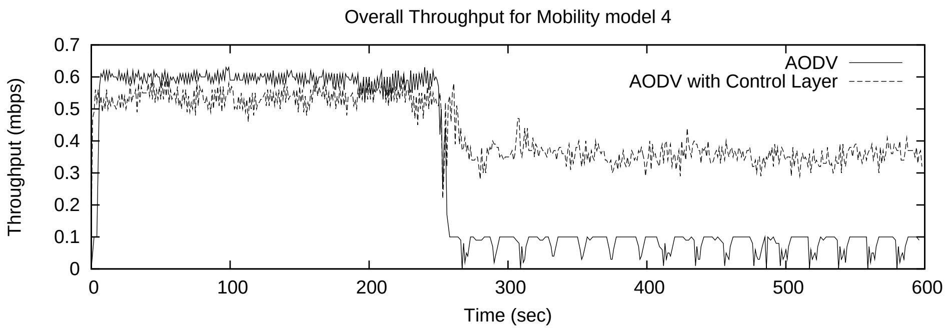

Figure 4. 19 Throughput of All Flows for Model 4

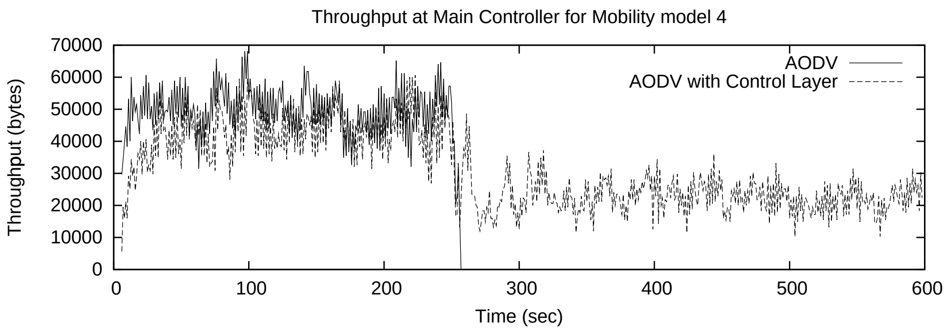

Figure 4. 20 Throughput at Main Controller for Model 4

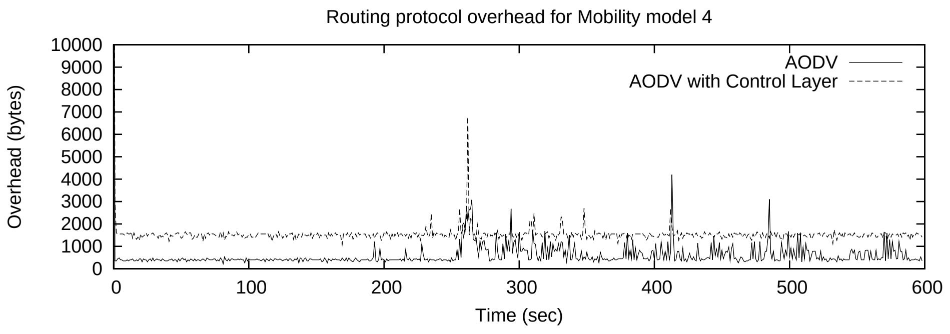

Figure 4. 21 Network Overhead (Routing and Hello Packets) for Model 4

when using base AODV. The cluster mobility behavior is represented by flows 1-56 and until 410 seconds, a connection exists between all the mobile nodes either directly or through other nodes in the same cluster. However at 410 seconds, the two clusters become independent and we only have inter-cluster communication. Flows 1-14 represent the CBR connections from the cluster with 2 nodes (that is the reason why we see many connections being broken at 410 seconds) and flows 15-56 represent the CBR connections from the cluster with 6 nodes (more flows remain connected for the entire duration of the simulation).

Like in the previous models, AODV with control layer algorithm ensures a steady connection of all the mobile nodes throughout the duration of simulation. Throughput graphs for this model are presented in Figures 4. 23 and 4. 24. Throughput at the base station drops to zero at 250 seconds, when using AODV, while AODV with control layer ensures a stable throughput. The overhead plot for this model is presented in Figure 4. 25

4.3.6 Simulation Model 6: Random Waypoint Model

Model 6 represents the Random Waypoint Model discussed in [44]. The mobility file was generated using the *setdest* tool distributed with the ns simulator. Each node selects a random destination and moves towards it, and on reaching the destination, pauses for the given *pause time* and then picks another destination and moves towards it. This pattern is repeated for the entire duration of simulation. Also, the speed of each node is different with respect to other nodes. The nodes can be expected to move in a manner as shown in Figure 4. 26, but it is to be noted that unlike other previous models this figure does not represent the actual node destination and direction. From the link stability graph shown in Figure 4. 27, it is evident that all the nodes remain connected throughout the duration of simulation when using both AODV and AODV with control layer algorithm. The performance of both the protocols are similar and can be verified from the throughput graphs in Figures 4. 28 and 4. 29. Also, like in previous models there is a very minimal increase in overhead, a mean increase of 1200 bytes (Figure 4. 30).

Thus, using AODV with control layer algorithm in models where the nodes stay within the transmission range and any of the existing protocols would ensure node connectivity, does not affect the performance. From the simulation results of the 6 models it is very evident that AODV with the

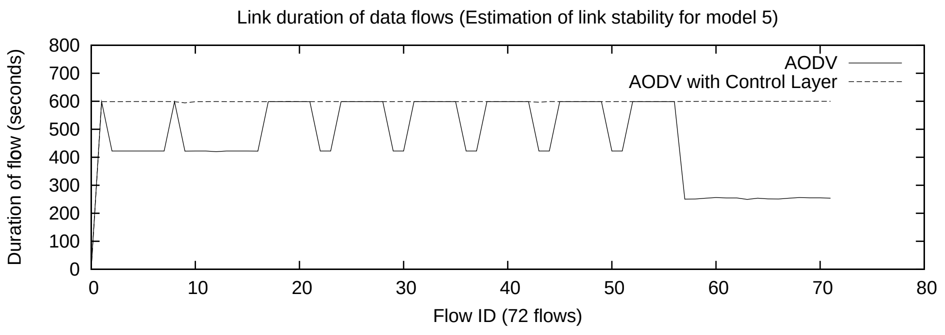

Figure 4. 22 Comparative Link Stability Analysis for Mobility Model 5

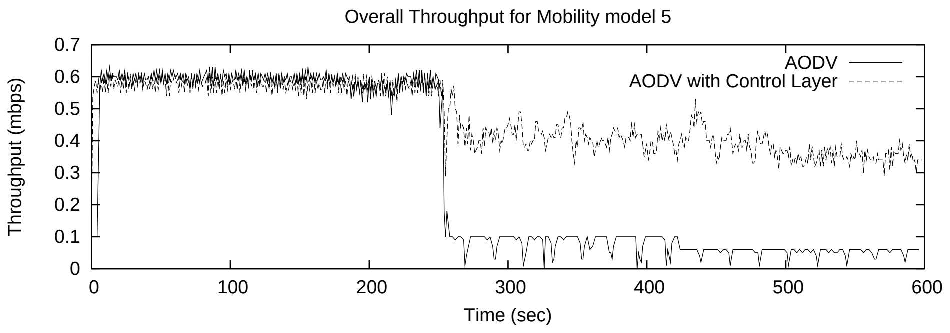

Figure 4. 23 Throughput of All Flows for Model 5

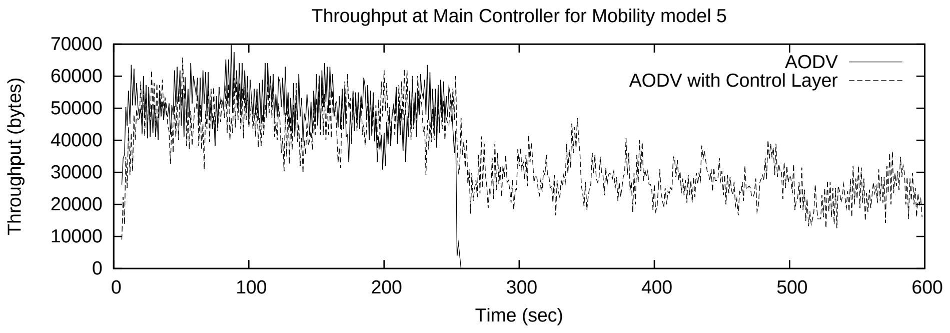

Figure 4. 24 Throughput at Main Controller for Model 5

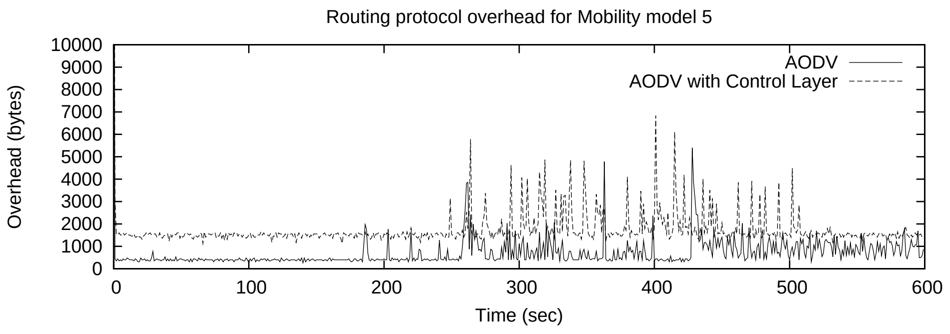

Figure 4. 25 Network Overhead (Routing and Hello Packets) for Model 5

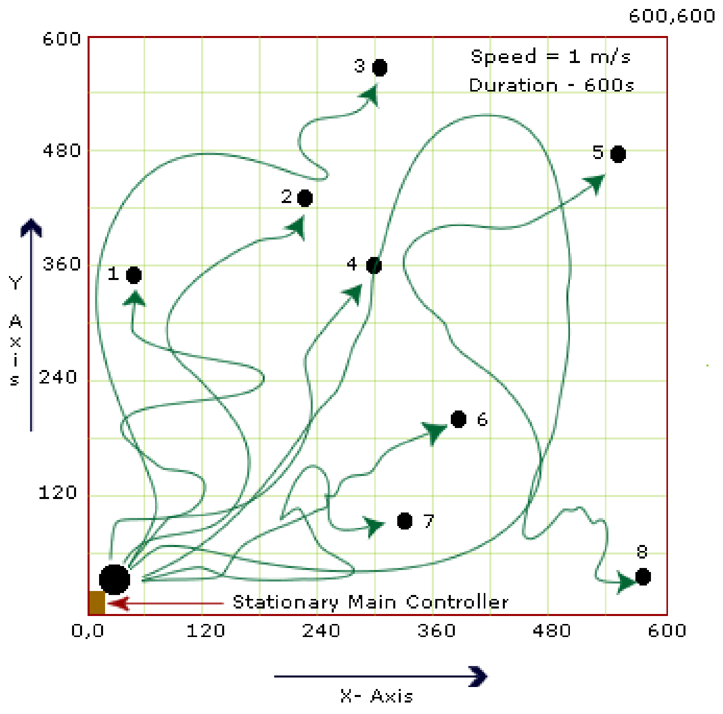

Figure 4. 26 Mobility Model 6 (Random Waypoint Model)

proposed control layer algorithm ensures consistent node connectivity of the entire network by making sure that all the nodes stay connected with the main controller either directly or through other nodes.

4.3.7 Area of Coverage

This section analyzes the performance of the proposed control layer algorithm for the distance covered by the mobile nodes when using the various mobility models. The distance traversed by the mobile nodes is estimated as the maximum distance at which the nodes can establish communication with the main controller. However, when using the base AODV algorithm, the nodes would still continue their mobility (even after reaching the transmission range of the main controller) until they reach their preset destination. But in the case of AODV with control layer algorithm, this measure (distance covered) is an indicator of the distance traversed towards the destination, at which the mobile nodes are stopped. Unlike mobility models 1 to 5, nodes in model 6 keep changing their destination and rate of mobility, and hence the values presented in Table 4.6 are an approximation of

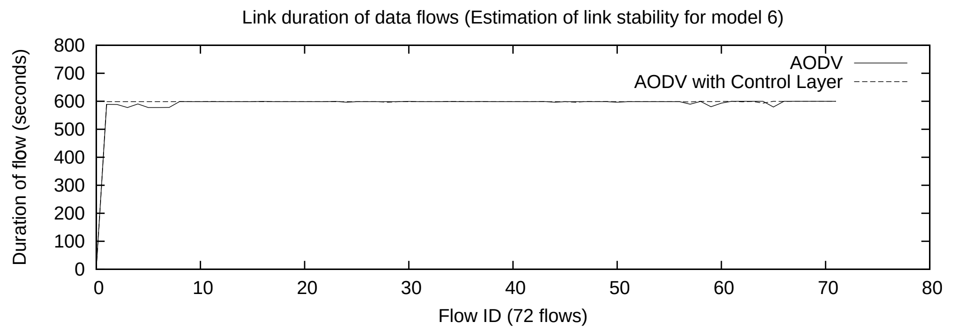

Figure 4. 27 Comparative Link Stability Analysis for Mobility Model 6

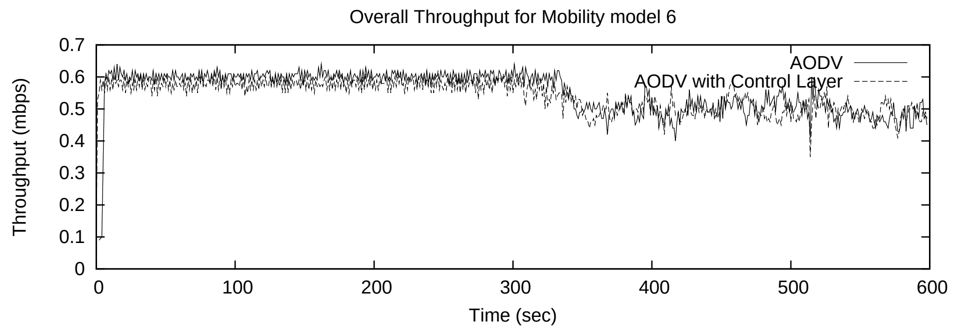

Figure 4. 28 Throughput of All Flows for Model 6

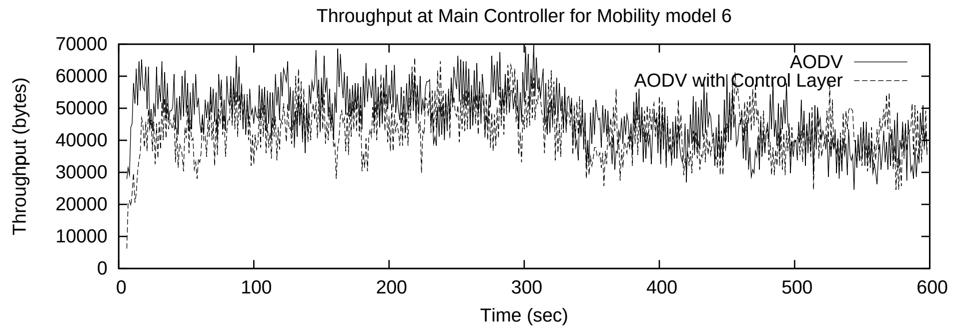

Figure 4. 29 Throughput at Main Controller for Model 6

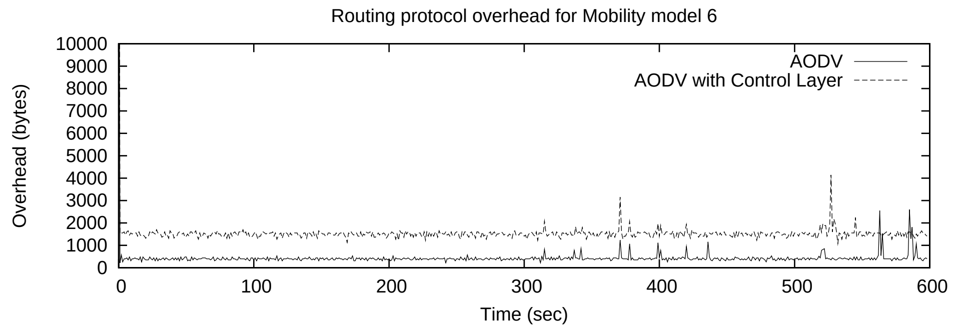

Figure 4. 30 Network Overhead (Routing and Hello Packets) for Model 6

Table 4.1 Comparative Distance Covered by Mobile Nodes in Model 1

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	256.03	127.81
2	255.93	248.65
3	255.77	368.48
4	256.38	486.13
5	256.07	569.49
6	246.20	595.96
7	250.02	592.83
8	256.20	559.12

Table 4.2 Comparative Distance Covered by Mobile Nodes in Model 2

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	253.41	126.86
2	241.85	243.14
3	253.51	360.58
4	250.59	471.12
5	255.66	136.25
6	251.65	251.68
7	198.20	373.46
8	255.58	326.03

the distance covered by the mobile nodes with respect to its final position at which it could establish a communication link with the main controller.

Table 4.1 compares the distance traversed by the mobile nodes in model 1 using base AODV and AODV with control layer algorithm (Section 4.3.1). When using the base AODV algorithm, all the mobile nodes break their communication with the main controller after traversing a distance between 246.2 meters and 256.03 meters. However, by using AODV with control layer algorithm, nodes 5-8 traverse a distance greater than 559.12 meters, with nodes 1-4 being stopped at equally spaced intervals to enable multi-hop routing and constant communication. Also, nodes 6 and 7 continue their mobility throughout the duration of simulation (600 seconds), while constantly communicating with the main controller. The proposed control layer algorithm clearly outperforms the base AODV protocol with respect to the distance covered by the nodes in mobility model 1.

The distance traversed by the mobile nodes in model 2 (Section 4.3.2) is shown in Table 4.2. The maximum traversed distance when using the base AODV algorithm is 255.7 meters by node 5, after which all the nodes break their communication with the main controller. However, when

Table 4.3 Comparative Distance Covered by Mobile Nodes in Model 3

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	249.00	116.62
2	197.92	173.01
3	230.12	153.26
4	241.48	161.86
5	249.49	183.00
6	249.82	193.78
7	249.38	203.60
8	248.92	213.43

Table 4.4 Comparative Distance Covered by Mobile Nodes in Model 4

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	251.62	124.04
2	255.24	243.75
3	255.10	356.39
4	256.25	441.14
5	256.53	349.85
6	254.19	337.13
7	255.90	276.59
8	255.13	380.04

using AODV with control layer algorithm, the maximum traversed distance without breaking the communication link with the main controller is 471.12 meters by node 4, while the minimum traversed distance is 126.86 meters by node 1. Again, for mobility model 2, the proposed control layer algorithm achieves greater area of coverage while ensuring constant communication with the main controller.

In Table 4.3 the distance traversed by the mobile nodes in mobility model 3 (Section 4.3.3) is compared. Unlike in the previous two cases, the distance traversed by the mobile nodes using the base AODV algorithm is higher than the distance achieved using AODV with control layer algorithm. This behavior is attributed to two significant factors: (a) mobility pattern and (b) *link threshold*. In mobility model 3 (Figure 4. 11), all the nodes move towards the periphery of a circle with the main controller at the center. Such a mobility pattern does not provide scope for multi-hop routing, and AODV with control layer algorithm stops each node at the *link threshold* to ensure constant communication with the min controller. Also, this work uses a *link threshold* of $3.652e^{-9}$ Watts, while the receiving threshold (RXThresh_)for the base AODV is set to $3.652e^{-10}$ Watts. Per-

Table 4.5 Comparative Distance Covered by Mobile Nodes in Model 5

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	253.05	127.05
2	254.26	241.90
3	256.32	200.49
4	256.36	313.63
5	256.30	433.37
6	256.20	394.29
7	250.87	504.07
8	255.90	449.26

Table 4.6 Comparative Distance Covered by Mobile Nodes in Model 6

Node ID	AODV (meters)	AODV with Control Layer (meters)
1	447.10	300.18
2	86.04	86.07
3	289.08	298.02
4	191.32	191.76
5	303.30	307.15
6	231.06	229.76
7	115.06	115.00
8	136.89	135.40

formance of the proposed control layer algorithm could be improved by increasing the *link threshold* value to the receiving threshold (RXThresh_).

Tables 4.4 and 4.5 compare the distance traversed by the mobile nodes when using base AODV and AODV with control layer algorithm, for mobility models 4 and 5 (Sections 4.3.4 and 4.3.5) respectively. In both cases, AODV with control layer algorithm results in larger distances being traversed by the mobile nodes. In mobility model 6, all the nodes (except for node 1) remain within the transmission range of the main controller and hence the distance traversed by the nodes when using base AODV is similar to the distance traversed by the nodes when using AODV with control layer algorithm. Thus, the proposed control layer algorithm results in significant increase in the distances being covered by the mobile nodes while ensuring constant *NODE CONNECTIVITY*.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis the idea of using ad hoc networking in support of USAR applications is proposed. A new control layer algorithm is designed to guarantee node connectivity and ensure steady throughput with a very minimal increase in overhead. Node connectivity is critical to the performance of a network where link breakages or node loss could incur heavy performance and financial damages, such as a network of autonomous robots collaborating in search and rescue operations. The proposed algorithm is implemented in ns-2.26 simulator, and its performance is analyzed using 6 different mobility models. The simulation results show that the control layer based algorithm guarantees node connectivity with very minimal increase in overhead.

5.2 Future Work

This is the first work attempting to ensure node connectivity for ad hoc networks and suggest the idea of a control layer based solution. In this work the proposed algorithm was verified and its usefulness and performance was evaluated. However, the idea of node connectivity and control layer based solution introduces a vast scope for future research work. For example, Based on the information collected at the main controller, a graphical tool could be designed to provide the rescue personnel with exact information on the position and status of the mobile nodes, stability of the links and also any data received from the nodes. This would also facilitate in tele-operating the robots. The proposed model uses an HELLO INTERVAL and UPDATE INTERVAL of 1 second, and the MONITOR INTERVAL is 1.5 seconds. It would be interesting to study the effect of varying these values on the performance of the system. It is essential to select the right value for these parameters, as a smaller interval leads to network congestion and increased overhead, while large intervals might to lead to stopping a node well after it breaks away from the network. The update packets sent by the mobile nodes have a field for sequence numbers which is not used in this work. However, the performance improvement achieved by using sequence number for update packets could be studied. A performance comparison of TCP and UDP for the proposed solution could be done, and also analyze the performance of various TCP versions for the proposed method. Also, instead of sending the update packets separately, they could be piggy-backed with the data packets. The performance of the proposed method at high mobility rates could be studied. The performance of the proposed method could be compared with existing ad hoc routing solutions that aim in selecting stable links for routing packets based on signal stability, location stability and energy of the nodes.

REFERENCES

[1] I. Chlamtac, M. Conti, and J. J.-N. Liu, “Mobile Ad hoc Networking: Imperatives and Challenges,” *Ad Hoc Networks*, vol. 1, pp. 13–64, July 2003.

[2] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad hoc Wireless Networks,” in *Mobile Computing*, Imielinski and Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353.

[3] J. A. Freebersyser and B. Leiner, *A DoD Perspective on Mobile Ad Hoc Networks*, 2nd ed., ser. Ad hoc networking, edited by Charles. E. Perkins. Addison-Wesley, 2001, vol. 1, chapter 2, pp. 29–51.

[4] “A short description on ad hoc networks,” http://www.ka9q.net/papers/wcnc99/.

[5] “New York Search and Rescue (FAQ),” http://newyorksearchandrescue.org/faq.html.

[6] R. Murphy, J. Casper, J. Hyams, M. Micire, and B. Minten, “Mobility and Sensing Demands in USAR,” in *Conference of the IEEE Industrial Electronics Society (IECON)*, Nagoya, Japan, October 2000.

[7] “Remote controlled Robots search World Trade Center rubble,” http://www.tms.org/pubs/journals/JOM/0112/News/News3-0112.html.

[8] “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” The Institute of Electrical and Electronics Engineers, New York, IEEE Std. 802.11-1997, July 1997.

[9] M. Gerharz, C. D. Waal, M. Frank, and P. Martini, “Link Stability in Mobile Ad Hoc Networks,” in *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN’02)*, Tampa, FL, November 2002, pp. 30–39.

[10] R. Dube, C. D. Rais, K.-Y. Wang, and S. K. Tripathi, “Signal Stability based Adaptive Routing (SSA) for Ad-Hoc Mobile Networks,” *IEEE Personal Communications*, vol. 4, pp. 36–45, February 1997.

[11] M. Schwartz and T. Stern, “Routing Techniques used in Computer Communication Networks,” *IEEE Transactions on Communications*, vol. COM-28, pp. 539–552, April 1980.

[12] J. M. McQuillan, I. Richer, and E.C. Rosen, “The New Routing Algorithm for the ARPANET,” *IEEE Transaction on Communications*, vol. COM-28(5), pp. 711–719, May 1980.

[13] J. Moy, “Open shortest path first protocol,” The Internet Engineering Task Force (IETF), RFC 2328, April 1998.

[14] C. Hendrick, “Routing information protocol,” The Internet Engineering Task Force (IETF), RFC 1058, June 1988.

[15] C. E. Perkins and P. Bhagwat, “Highly dynamic Destination Sequenced Distance-vector Routing (DSDV) for Mobile Computers,” in *Proceedings of the ACM SIGCOMM Symposium on Communication, Architecture and Protocols*, vol. 24, October 1994, pp. 234–244.

[16] S. Murthy and J.J. Garcia-Luna-Aceves, “A Routing Protocol for Packet Radio Networks,” in *Proceedings of the ACM First International Conference on Mobile Computing and Networking (MOBICOM ’95)*, November 1995, pp. 86–95.

[17] C. E. Perkins, Belding-Royer, and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” The Internet Engineering Task Force (IETF), RFC 3561, July 2003.

[18] V. D. Park and M. S. Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks,” *INFOCOM*, pp. 1405–1413, April 1997.

[19] C.-K. Toh, “Associativity-Based Routing for Ad Hoc Mobile Networks,” *Wireless Personal Communications*, vol. 4-2, pp. 103–139, March 1997.

[20] Y.-B. Ko and N. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks,” in *Proceedings of the Fourth ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom’98)*, Dallas, Texas, October 1998, pp. 66–75.

[21] J.J. Garcia-Luna Aceves; M. Spohn, “Source-tree routing in wireless networks,” in *Proceedings of the Seventh International Conference on Network Protocols (ICNP ’99)*, October 1999, pp. 273–282.

[22] D. Camara and A. A. Loureiro, “A Novel routing algorithm for ad hoc networks,” in *Proceedings of the 33rd International Conference on System Sciences*, Maui, Hawaii, January 2000.

[23] I. Stojmenovic and X. Lin., “Power-aware localized routing in wireless networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, pp. 1122–1133, October 2001.

[24] Z. J. Haas and M. R. Pearlman, *Zone Routing Protocol (ZRP) A Hybrid Framework for Routing in Ad hoc Networks*, 2nd ed., ser. Ad hoc Networking, edited by Charles. E. Perkins. Addison-Wesley, 2001, vol. 1, chapter 7, pp. 221–253.

[25] C. K. Toh, *Ad Hoc Mobile Wireless Networks*. Upper Saddle River, New Jersey: Prentice Hall, 2002.

[26] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, “A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols,” *Mobile Computing and Networking*, pp. 85–97, 1998.

[27] S. R. Das, C. E. Perkins, and E. E. Royer, “Performance comparison of two on-demand routing protocols for ad hoc networks,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, March 2000, pp. 3–12.

[28] 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 the 5th annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, Washington, August 1999, pp. 195–206.

[29] A. Boukerche, “A Performance comparison of routing protocols for ad hoc networks,” in *Proceedings of the IEEE Parallel and Distributed Computing for Wireless and Mobile Systems, LNCS*, Springer-Verlag, Berlin, August 2001, pp. 195–206.

[30] M. Mauve, J. Widmer, and H. Hartenstein, “A survey on position-based routing in mobile ad hoc networks,” *IEEE Network Magazine*, vol. 15(6), pp. 30–39, November 2001.

[31] S. Giordano, I. Stojmenovic, and L. Blazevie, “Position based routing algorithms for ad hoc networks: a taxonomy,” 2001.

[32] G. Holland and N. Vaidya, “Analysis of TCP Performance over Mobile Ad Hoc Networks,” in *Proceedings of the 5th annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, Washington, 1999, pp. 219–230.

[33] Venkatesh Ramarathinam and Miguel Labrador, “Performance analysis of TCP over static ad hoc networks,” in *Proceedings of the ISCA 15th International Conference on Parallel and Distributed Computing Systems (PDCS)*, September, 2002, pp. 410–415.

[34] P. Sholander, A. Yankopolus, P. Coccoli, and S. Tabrizi, “Experimental Comparison of Hybrid and Proactive MANET Routing Protocols,” in *Proceedings of the Military Communications Conference (MILCOM)*, October 2002.

[35] J. Raju and J. Garcia-Luna-Aceves, “A Comparison of On-demand and Table-driven routing for Ad hoc wireless Networks,” in *Proceedings of the IEEE International Conference on Communications*, JUNE 2000, pp. 1702–1706.

[36] A. Winfield and O. Holland, “The application of wireless local area network technology to the control of mobile robots,” *Journal of Microprocessors and Microsystems (Elsevier), Vol 23/10*, pp. 597–607, August 2000.

[37] A. Winfield, “Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots,” *Distributed Autonomous Robotic Systems*, pp. 273–282, 2000.

[38] “DARPA project develops compact ad hoc networking radios,” *Robotics Update*, vol. 3, no. 1, pp. 1122–1133, April 2003.

[39] K. V. Kevin Fall, *The NS Manual (formerly ns Notes and Documentation)*, the VINT Project A collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC.

[40] “NAM: Network Animator,” http://www.isi.edu/nsnam/nam/.

[41] I. Stojmenovic, “Position-based routing in ad hoc networks,” *IEEE Communications Magazine*, vol. 40, no. 7, pp. 128–134, July 2002.

[42] Lucent Technologies, “WaveLAN/PCMCIA card user’s guide,” October 1996.

[43] Theodore S. Rappaport, *Wireless Communications: Principles and Practice*. New Jersey: Prentice Hall, 1996.

[44] T. Camp, J. Boleng, and V. Davies, “A Survey of Mobility Models for Ad Hoc Network Research,” *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[45] D. Johnson and D. Maltz, *Dynamic Source Routing (DSR) in Ad hoc Wireless Networks*, 2nd ed., ser. Mobile Computing. Kluwer Academic Publishers, 1996, pp. 153–181.