UNIVERSITÀ DEGLI STUDI DI CATANIA
FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI
DOTTORATO DI RICERCA IN INFORMATICA

DISTRIBUTED ALGORITHMS FOR THE CONFIGURATION AND MANAGEMENT OF WIRELESS AD-HOC NETWORKS WITH MOVABLE BASE STATIONS CONSTRAINED BY MOTION AND COMMUNICATION OBSTACLES

SALVATORE CRISTALDI

A dissertation submitted to the Department of Mathematics and Computer Science and the committee on graduate studies of University of Catania, in fulfillment of the requirements for the degree of doctorate in computer science.

ADVISOR
Prof. Alfredo Ferro

COORDINATOR
Prof. Domenico Cantone

XXIII CICLO

I wish to thank my wife, Rosa.
She has been patient, raised me, supported me,
taught me, and loved me.
To her I dedicate this thesis.

Acknowledgements

It is difficult to express my gratitude to my advisor, Prof. Alfredo Ferro, for his unstinting commitment to help me to see this project through to its final completion, and his equally generous and wise guidance during its development.

I am also extremely grateful to Dr. Alfredo Pulvirenti, Dr. Rosalba Giugno and Dr. Giuseppe Pigola. They have helped and supported me through the last three years.

Abstract

In this thesis, we propose a protocol for dynamic reconfiguration of ad-hoc wireless networks with movable base stations in presence of obstacles. Hosts are assigned to base stations according to a probabilistic throughput function based on both the quality of the signal and the base station load. In order to optimize space coverage, base stations cluster hosts using a distributed clustering algorithm. Obstacles may interfere with transmission and obstruct base stations and hosts movement. To overcome this problem, we perform base stations repositioning making use of a motion planning algorithm on the visibility graph based on an extension of the bottleneck matching technique.

We implemented the protocol on top of the NS2 simulator as an extension of the AODV. We tested it using both Random Way Point and Reference Point Group mobility models properly adapted to deal with obstacles.

Experimental analysis shows that the protocol ensures the total space coverage together with a good throughput on the realistic model (Reference Point Group) outperforming both the standard AODV and DSR.

Results of this thesis have been published in the journal Ad-Hoc Networks [14].

Contents

Chapter 1

Introduction

Many wireless systems rely on fixed base stations organized in a backbone of wired links. Base stations are special nodes having more resources (processing power, memory capacity, energy supply, etc.) than the mobile hosts. Base stations provide connectivity and other services for mobile hosts and are connected through bidirectional links according to some topological structures such as trees.

In several applications however, such as military or emergency operations, wired networks may be not available or not suitable to guarantee communication [23, 46]. In those situations, Mobile Backbone Wireless Networks (MBWN) [25] can ensure communication. MBWN are wireless networks with a backbone of movable base stations. A crucial issue in MBWN models is the network organization and management since a simple flat topology could result unfeasible because of the huge overhead created by the network traffic. A typical MBWN organization relies on topology aggregation in which hosts are grouped and communicate through hierarchical control strategies. This yields models which scale well with network dimension maintaining control on important features such as code separation among hosts clusters, channel access, routing, power control, and bandwidth allocation [23]. Clustering plays an important role since it provides a convenient organization of the network possessing several advantages: routing overhead reduction, spatial reuse of shared channel together with a simple and feasible power control mechanism [33].

In MBWN host clustering can be exploited by sequential or distributed algorithms to organize the host topology and to efficiently move base stations maintaining connection. Mobile base stations are initially located in the centers of the clusters. Then they periodically move to the next clustering centers trying to minimize the energy consumption. In absence of motion obstacles, repositioning can be performed by bipartite matching algorithms which try to minimize total and/or maximum distances [20]. When obstacles are present, matching algorithms have to be properly adapted to deal with them as in a motion planning strategy.

A key component of a MBWN protocol is the throughput optimization. An acceptable solution tries to find a good balance between the closeness to the assigned base station and the load of each base station.

We propose a model, integrated into a communication protocol, for the distributed dynamic reconfiguration of Event-Driven Mobile Backbone Wireless Networks (EDMBWN) [25] in presence of obstacles. EDMBWN are special MBWN in which base stations move when certain events happen following some scheduling or special triggers.

Our protocol consists of two main phases:

- **Space coverage optimization**: The space coverage is optimized by using a distributed version of the Antipole Clustering algorithm [11] which identifies suitable base stations positions (clusters' centroids). The distributed clustering is guided by a Euclidean Minimum Spanning Tree (EMST) which represents the base station clustering backbone.

  Obstacles restrict base stations and hosts movement and may interfere with transmission. Repositioning is performed by a fast motion planning algorithm based on the bottleneck matching algorithm [1] on visibility graphs [47].

- **Throughput optimization and network maintenance**: The hosts are periodically assigned to base stations according to a probabilistic throughput function. The throughput combines the quality of the signal (inversely proportional to the distance) and the potential load of the base stations due to the number of hosts in their neighborhood. Each host computes such a function yielding a score which allows to identify the best base station to join with. Notice that, since we use a single communication channel (802.11x protocol) the throughput achieved by a host transmitting to its assigned base station is influenced by the totality of the hosts in the same neighborhood and not only by those joint with the base station [48]. Base stations communication is ensured by the standard AODV (Ad-hoc On-Demand Distance Vector [40]).

The model has been implemented on top of the Network Simulator NS2 [38, 27] and it has been tested using two different mobility models: the Random Waypoint (RWPM) [30] and the Reference Point Group (RPGM) [26]. The former is widely used but unrealistic due to the randomness of hosts movement. In the RPGM hosts collaborate in small groups and their movement follows the group they belong to (i.e. rescue units, platoons of soldiers, etc.).

In order to model more complex situations, we adapted RWPM and RPGM to manage obstacles interfering with both movement and communication. Nodes turn around obstacles when they obstruct their way and a signal attenuation factor is taken into account.

Furthermore to consider a scenario in which base stations are overloaded, a skewed traffic distribution is tested.

Experimental analysis shows that the proposed protocol equipped with RPGM in presence of obstacles ensures the total space coverage. In this situation it outperforms the standard AODV and DSR. On the other hand if RWPM is used a slightly lower performance is shown. The usage of a throughput function improves packets delivery compared to the simple closest base station assignment.

Chapter 2

Background and Related Work

2.1 Ad-hoc networks

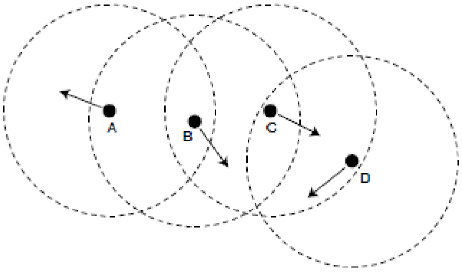


Figure 2.1: Example of a wireless ad-hoc network

In wireless networks, mobile nodes can communicate in different ways. In a centralized approach nodes communicate through stationary base stations which represent a backbone of the network. In a decentralized approach nodes cooperate each other for the communication and no base station backbone is present. Generally we refer to this kind of networks as ”ad-hoc” networks. Figure 2.1 shows a simplified ad-hoc network with 4 nodes. At a certain time, each node has a transmission range outlined with a dotted circle and a movement direction shown by a arrow. Node A wishes to communicate with node D. Since node A can not directly communicate with node D, it needs to route packets through the intermediate nodes B and C.

2.2 Ad-hoc routing protocols

In order to communicate, mobile nodes need a routing protocol. Protocols are typically classified in two different classes. Reactive approach (source initiated, on-demand driven) and proactive approach (table driven). There are also hybrid routing protocols that integrates both routing strategies. A typical weakness of reactive and proactive routing protocols is the scalability: when the network size increases, the number of packets exchanged for the communication have a huge growth. For this reason these types of protocols may be not suitable for many realistic network conditions. In real scenarios with several hundreds or thousands of nodes, studying the behavior of hybrid protocols would be interesting. In what follows, proactive and reactive strategies will be briefly described and two of the most important protocols (AODV and DSR) for each strategy will be analyzed.

2.2.1 Proactive approach

Proactive protocols attempt to maintain routes from each node to all other nodes in the network [43]. This protocols are also called table driven protocols since they need to maintain tables for storing routing information. Whenever a change in network topology occurs these changes are propagated through the network by the means of broadcast or flooding. These updates are vital in order to maintain a consistent view of the network topology.

2.2.2 Reactive approach

Reactive protocols create routes upon requests [43]. When a node requests a route toward another node a route discovery process is started in the network. The route discovery ends once a route is found or when all possible routes are examined. Then, the discovered route will be maintained until it is no longer valid or not desired.

2.3 AODV

Ad-hoc On-Demand Distance Vector routing protocol was first proposed in [40]. AODV is based on a routing protocol called DSDV and described in [39]. AODV limits the number of broadcasts by requesting routes when needed only as opposed to DSDV that requires a continuous routes updating [43]. In AODV, nodes that are not present in a selected path do not maintain routing information nor do they participate in any periodic routing table exchanges. Nodes can be made aware of their neighbors in two ways. When a node receives a broadcast from another node, that is currently not known, it includes this neighbor in its local connectivity information. However if a node has not been active in the ad-hoc network, it can make its neighbors aware of its presence by broadcasting a HELLO message, These broadcast are periodically done. The HELLO messages are specified to use a time to live (TTL) value of 1, which means that the message will only be broadcasted one hop away. When a source node wishes to communicate with another node and it lacks a valid route to that node, a path discovery process is initiated. In the path discovery process the source node broadcasts a route request (RREQ) packet to its neighbors. The neighbors reply with a route reply (RREP) packet if they have a valid route to the destination node otherwise they broadcasts a RREQ packet to their neighbors. Figure 2.2 shows how a RREQ packet is propagated trough the network from the source node toward the destination node. In this figure it is assumed that no intermediate nodes know the route from source to destination.

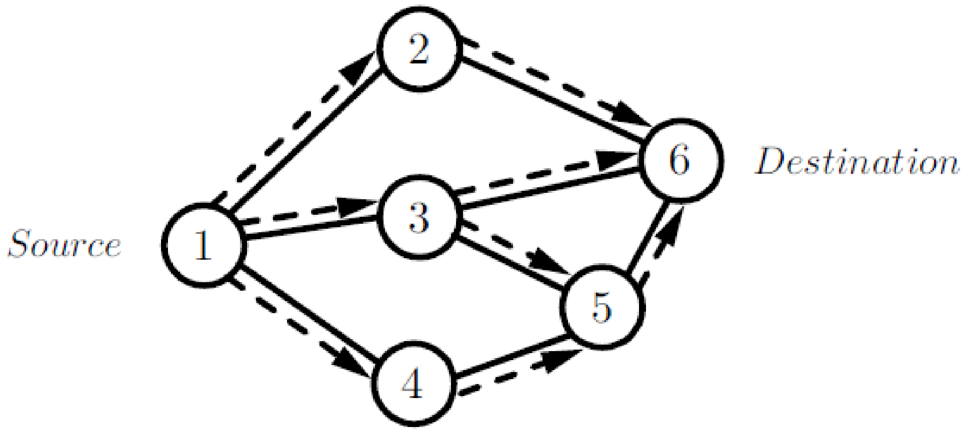


Figure 2.2: Propagation of a RREQ packet through the network.

Every time a new RREQ packet is broadcasted from the source, a sequence number called broadcast_id is incremented. If a node receives a RREQ packet with a sequence number that is less or equal to a previous received RREQ packet, it drops the packet. A RREQ packet has two other sequence numbers in addition to broadcast_id; source sequence number and destination sequence number. The source sequence number indicates how fresh the route information is for the reverse path to the source. The destination number which specifies how fresh a route to the destination is. When a RREQ packet is broadcasted through the network, a reverse path is built from all nodes back to the source. When a intermediate node has a valid route entry toward the destination, it compares the destination sequence number in the RREQ packet with the destination sequence number available in its own routing entry. If the destination sequence number in the RREQ packet is greater than the destination sequence number in the routing entry then it must not respond with a RREP. This is done to avoid outdated information to be transmitted. On the other hand the intermediate node should rebroadcast the RREQ packet. When the routing entry in the intermediate node has a greater or equal destination sequence number with respect to the one in the RREQ packet, it unicasts a RREP packet back to the node from which it received the RREQ packet. Once a route toward the destination have been found a reverse path to the source is established. A RREP packet from the

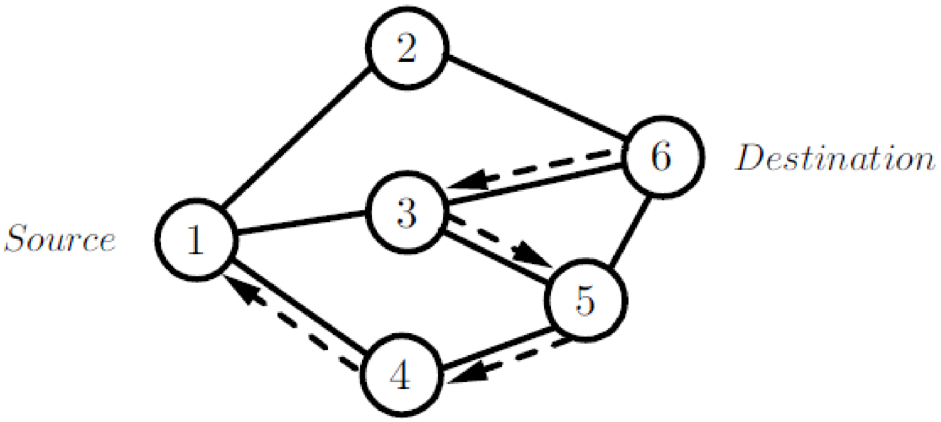


Figure 2.3: Propagation of a RREP packet.

destination node or from a intermediate node will then be unicasted along the reverse path and all intermediate nodes updates their routing entries to include the node from which the RREP came. Figure 2.3 shows the route for the RREP packet. This route is in fact the reverse path established from the propagation of the RREQ packet.

2.4 DSR

The Dynamic Source Routing protocol was proposed by Maltz et. al in [31]. DSR is a reactive protocol which uses source routing [43]. When a node wishes to send a packet to another node it includes a source route in the packets header. This source route is a sequence of nodes addresses which the packets must traverse in order to reach the destination. The packet is sent to the first hop. If a node receives a packet and it is not the final destination, it just forwards the packet to the next hop in the source route. Each node maintains discovered routes in a cache. Whenever a node wants to send a packet, it consults its route cache for a route. If a source route to the destination exists, then the node can proceed as described above. However if no source route toward the destination can be found in the route cache, a route discovery process is initiated.

In the route discovery process the source node broadcasts a route request packet. Every route request contains source address, destination address and a unique identification number. If a intermediate node lacks a route to the destination it adds its address to the route record of the packet and forwards it on its outgoing links. Figure 2.4 shows how a route request is propagated

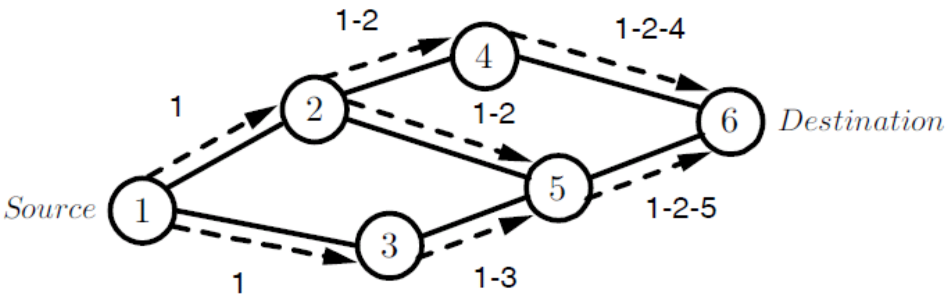


Figure 2.4: Propagation of a route request and building of the source record.

through the network. Once a route request has reached the destination or an

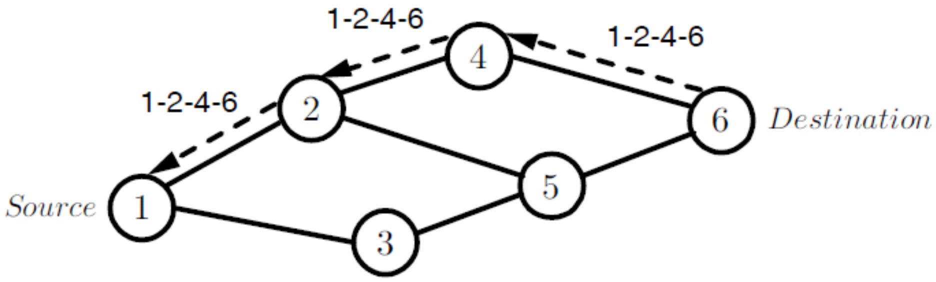


Figure 2.5: Propagation of route reply by using the reversed source record.

intermediate node which knows a route to the destination, a route reply is generated. When the destination node generates the route reply it adds the information from the route record into the route reply. If an intermediate node generates the route reply it appends the route record with the route found in the route cache and adds it to the route reply. If symmetric links are supported the route reply uses the reversed route found in the route record. Otherwise the route reply packet can be piggybacked on a route request packet. Figure 2.5 shows the propagation of a route reply using the reversed source record. Once a route has been establish it needs maintenance and for this purpose route error packets are generated at link failures. All nodes who receive a error packet removes the failed hop from the route cache and truncate all routes, containing this hop, to the previous hop. One of the advantages DSR has is that it can use promiscuous listening to learn about routes without transmitting any control packets. It works as follows; if a node a receives a packet that is targeted for a different node it can check whether a shorter route exists by sending through the node itself. In such a case a route reply is sent to the source identified in the source route. Route discovery in DSR is similar to AODV but there are some differences. DSR packets need to carry a complete source record while AODV packets only need to contain destination address which will lead to a lower overhead. The downside with AODV is that it requires symmetric links instead of DSR.

2.5 Topology Maintenance in Wireless Networks

Network topology is generally managed through hosts space clustering. When base stations are not present some of the hosts are elected as special nodes to guarantee intra and inter clusters communication. In the presence of base stations, hosts are clustered and each cluster is equipped with a base station. Inter cluster communication is ensured through a backbone. In these models hosts clustering is performed in a distributed fashion since a centralized algorithm would cause heavy overhead in network knowledge collection. Closest-base clustering strategy may not guarantee good network reliability.

For this reason, throughput functions trying to balance base station closeness and load are used. Topology management and routing in wireless networks mostly relies on clustering algorithms. Some clustering algorithms have been developed for "pure" ad-hoc networks in which hosts can communicate with or without the presence of base stations.

Gerla et al. in [23] present a radio network architecture which uses a revisited lowest-ID algorithm [17]. Each node has a distinct ID and it periodically broadcasts the list of nodes that it can hear (including itself). A node which can hear only nodes with ID higher than its own ID is a clusterhead. The remaining nodes choose their clusterhead to be the node of lowest-ID among those they can hear. Clusterheads are not linked through a backbone. Inter-cluster routing is realized by gateways: nodes that can hear two or more clusterheads. Moreover, in order to make the network more reliable, the notion of distributed gateway is introduced. In [33], Kwon and Gerla extended the Lowest-ID based algorithm by power and interference control mechanisms. Each clusterhead adjusts the signal level in order to minimally guarantee correct intracluster communication.

Lin and Gerla in [34] propose a two-hop distributed clustering algorithm without clusterheads. Nodes are divided into small groups with two-hop intra-cluster communication as in [23]. Communications across clusters are guaranteed through *repeaters* which are nodes that can communicate with nodes of different clusters. They also introduced a bandwidth routing algorithm based on Destination Sequenced Distance Vector (DSDV) for multimedia applications. The goal is to find the shortest path such that the free bandwidth is above the minimum requirement. When links fail because of mobility, then the routing algorithm is capable of finding new routes maintaining secondary paths.

Basagni in [5] presents two distributed clustering algorithms for ad-hoc networks. The first (Distributed Adaptive Clustering) is suitable for low mobility ad-hoc networks. Nodes grouping and clusterheads selection follow a weight-based criterion. The second algorithm (Distributed Mobility-Adaptive Clustering) is designed for high-mobility networks. Each node dynamically decides its role (clusterhead, ordinary) on the basis of the local network topology.

Gerla et al. in [22] propose a passive clustering in which a clusterhead is the node that sends a data packet first. Nodes in the radio coverage of this clusterhead are aggregated to it.

McDonald and Znati [36] propose an event-driven distributed clustering algorithm to aggregate nodes according to node mobility in order to balance the tradeoff between proactive and demand-based routing. A path connecting each pair of nodes in a cluster exists with a time-depending probability.

Alzoubi [3] presents two distributed heuristics to construct an approximate minimum connected dominating set used as a virtual backbone for routing in a wireless network. The topology is modeled as a unit disk graph [12]. This is a geometric graph in which there is an edge between two nodes if and only if their distance (number of hops) is at most one.

Banerjee and Khuller in [4] propose an algorithm to organize wireless nodes in clusters with a set of properties. Even though the program has been explicitly written for wireless sensors networks, it can be easily adapted to mobile networks in which nodes are quasi-static. The algorithm creates different layers in which it builds a set of clusters.

In [49] Kaixin and Gerla observe that in large ad-hoc networks a flat topology is a bottleneck for the performance. A two level hierarchical ad-hoc network with two independent routing protocols is introduced. A clustering technique based on random timers is used to partition hosts into small groups. Any node not belonging to a cluster starts to build a new cluster by sending a packet to claim itself as a clusterhead. All neighbors become members of the new cluster. The random timer is used to reduce conflicts. A modified AODV protocol has been used in order to keep hierarchy into account.

Foudriat et al. in [21] present a hierarchical network in which the lowest level is the cluster served by a mobile base station. At the top of the hierarchy there is a *leader* base station that is dynamically elected by the others. The leader base station periodically broadcasts the network topology. It is assumed that at least one base station has sufficient transmitting power to be heard by all clusters.

Lu et al. [35], propose a multilevel hierarchical mobile wireless network with movable base stations. The network is modeled as a tree in which mobile nodes are organized in hierarchical groups and each of them has a representative base station.

In [24] Gerla et al. present a hierarchical model similar to passive clustering for high speed wireless mobile backbone networks. The network is designed for battlefield operations in which unmanned flying nodes are organized in clusters. At low altitude clusters perform combat missions, whereas at mid altitude nodes execute surveillance and reconnaissance operations. The highest altitude nodes provide the connectivity. In order to perform routing the LANMAR protocol is adapted to deal with movable base stations.

In [45], Srinivas et al. focus on minimizing, placing and mobilizing the base stations in a mobile backbone network under two constraints: each host is covered by a base station and the backbone is connected. Base stations placement has been formulated in terms of Connected Disk Cover problem.

In [44], Srinivas and Modiano address the joint problem of placing a fixed number of mobile base stations in the plane assigning each host exactly one base station. The assignment is modeled by maximizing the minimum or the total throughput.

2.6 Mobility Models

The performances of a wireless network protocol are strictly related to the movement of its actors [10]. Due to the lack of real data, several mobility models have been designed to capture different type of realistic actor behaviors. In [10], authors give a survey on mobility models together with their impact on the performances of protocols.

One of the most simple and widely used mobility model is the Random Walk [16] (RWM). In RWM each host carries a speed and a direction. This information is randomly updated after a random interval of time. Boundaries are present in the simulation. When hosts reach the boundary they continue to move in the opposite direction with respect to the incoming angle. The model is unrealistic since it produces sharp turns, sudden stops and it tightly depends on the time parameter. The Random Waypoint Mobility

Model (RWPM) [30] is a slight modification of the RWM. Each mobile node randomly chooses a speed and a destination in the simulation area. When the node reaches the destination point, it pauses for a while (randomly chosen) and then restarts. Notice that, RWPM is RWM when the sleep time is set to zero. Sharp turns and sudden stops are avoided in [7], where directions and speed values are computed according to RWPM. This allows a smoother nodes movement. In [10] the elimination of sharp turns and sudden stops is managed through a mixed Gauss-Markov model.

In [26, 37, 26] hosts are grouped together in order to reflect relationship among them and they move according to the trajectory of their group. In the Reference Point Group mobility model [26] (RPGM) each group has a logical center, called Reference Point (RP) which defines the entire group motion behavior. In each group, global speed and direction are assigned to the RP. In addition local speed and direction are assigned to each host. In order to move a host (assigned to a RP), first the RP moves according to a motion vector, then a new host position is generated by adding a random motion vector to the new RP.

Authors in [28, 29] deal with obstacles. They observe that in the real world, movement patterns do not follow a random model, they tend to select a specific destination and follow a defined path to reach the destination, avoiding obstacles to the movement. Movement paths are realized through Voronoi [41] diagram of obstacle vertices. Hosts are placed across the paths. They move towards a destination using shortest paths. Intuitively, pathways tends to lie among adjacent obstacles and Voronoi diagrams of obstacles capture exactly this concept. Performance of the AODV [40] protocol is highly influenced by the associated mobility model. Indeed, using the above realistic protocol negatively affects efficiency with respect to the unrealistic random models.

In [32] Kim et al. define a mobility model based on real traces. Synthetic pathways generated by the model are compared with the real one showing a median relative error of 17%. Although the mobility model is more realistic than others, it is based on a single real trace data (the Dartmouth college). Furthermore its usage relies on the availability of real traces which are generally difficult to be known.

Chapter 3

Space Coverage Optimization

One of the main goals of a wireless network with movable base stations is to minimize uncovered areas. A host is covered by a base station if a bidirectional transmission link can be established. Consequently, each covered host has a base station within its transmission range (see Figure 3.1). Clustering

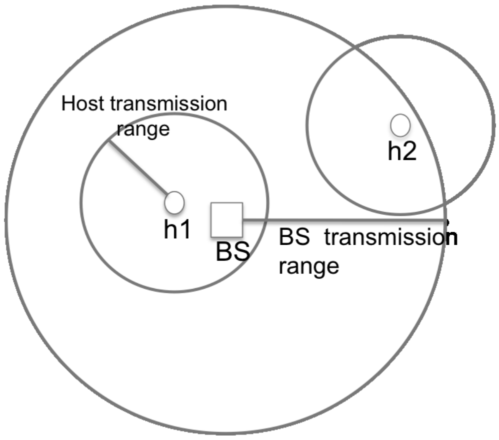


Figure 3.1: Transmission ranges: host h1 is covered, h2, although is in the transmission range of BS it will not be able to communicate since BS is not in its transmission range.

techniques allow to optimize space coverage by locating base stations in the clusters' centroids. Hosts are periodically clustered and each base station moves to an assigned centroid of a cluster. Two scenarios are possible: (i) if a base station knows all the hosts and the base stations positions, clustering can be performed by a centralized algorithm; (ii) if each base station has only a local knowledge, a distributed clustering is needed. If obstacles are present, they obstruct base stations and hosts movements and may also interfere transmission. In this situation, base stations repositioning is performed by a bipartite matching algorithm properly adapted to deal with obstacles as shown below. Finally, good performances of clustering and repositioning algorithms should be guaranteed during the evolution of the network. In this thesis we use an efficient hierarchical clustering algorithm Antipole Tree Clustering properly adapted to this scenario. In section 3.1, we sketch the sequential version of Antipole Tree Clustering in starred networks [20]. Next, in section 3.2, we propose a distributed version of the Antipole Tree Clustering and its application to MBWN. In section 3.3 we introduce a fast motion planning model to reposition base stations in the presence of obstacles.

3.1 The Antipole Clustering in Mobile Wireless Network with Starred Backbone

In this section, we review the sequential Antipole Clustering algorithm applied to EDMBWN organized as a starred network [20]. In this model the center of the star computes both clusters and centroids. Base stations are repositioned according to a fast *closest match* algorithm which minimizes both total and maximum distance [20].

Notice that if the number of base stations is different from the number of clusters the matching algorithm can be still applied by merging clusters or switching off the unused base stations.

The Antipole Tree Clustering algorithm (see Fig.3.2) is based on the observation that distant elements lie in different clusters. The algorithm finds a pair of distant element A, B (Pseudo-Diameter) in linear time with an approximation ratio of $1/\sqrt{2}$ with respect to the exact Diameter. It partition elements according to their proximity to one of the endpoints A, B. This top-down recursive splitting procedure will produce a binary tree whose leaves are the final clusters. Assume that a cluster radius σ, which guarantees good communication between each host and its base station, is given. The Antipole clustering of bounded radius σ [11], starting from a given finite set of points S, checks if the Pseudo-Diameter is greater than $2 \times \sigma$ (*splitting condition*). If this is not the case then splitting is not performed and the given subset

```
2D_ANTIPOLE_CLUSTERING($Tree$,$S$,$\sigma$, $Cluster\text{-}Centers$)

$Diag \longleftarrow$ 2D_DIAGONAL($S$, $Q$, $Pseudo\text{-}Center$);
{A,B} $\longleftarrow$ $Q$; // $Q$ contains the pseudo-diameter endpoints
if $Diag \leq 2\sigma$ then // splitting condition fails
  $Cluster - Centers \longleftarrow Cluster\text{-}Centers \cup \{Pseudo\text{-}Center\}$;
  return;
end if;
$Tree.Diagonal \longleftarrow Diag$;
$Tree.Center \longleftarrow Pseudo\text{-}Center$;
$S_l \longleftarrow \{O \in S | dist(O, A) < dist(O, B)\}$;
$S_r \longleftarrow \{O \in S | dist(O, B) \leq dist(O, A)\}$;
2D_ANTIPOLE_CLUSTERING ($Tree.left$, $S_l$, $\sigma$, $Cluster\text{-}Centers$);
2D_ANTIPOLE_CLUSTERING ($Tree.right$, $S_r$, $\sigma$, $Cluster\text{-}Centers$);
return;
END 2D_ANTIPOLE_CLUSTERING.
```

Figure 3.2: Euclidean 2-dimensional Antipole Algorithm.

is a cluster. Otherwise, the set is partitioned by assigning each point of the splitting subset to the closest endpoint of the Pseudo-Diameter $\{A, B\}$. In the plane this procedure can be efficiently performed in the following way. At each step, let T be the splitting subset to be processed, and let (P_{X_m}, P_{X_M}) and (P_{Y_m}, P_{Y_M}) be the four points of T having minimum and maximum Cartesian coordinates. Notice that, these four points belong to the convex hull of T. The diameter of these four points is the Pseudo-Diameter of T. Moreover the splitting condition for T is $\sqrt{(P_{X_m}.x - P_{X_M}.x)^2 + (P_{Y_m}.y - P_{Y_M}.y)^2} \geq 2 \times \sigma$, where $P.x$ and $P.y$ are the coordinates of the point P. If the splitting condition is satisfied then the diagonal of the rectangle is greater or equal than $2 \times \sigma$. Since the Pseudo-Diameter is at least $Diagonal/\sqrt{2}$, this yields that $\frac{Pseudo-Diameter}{2 \times \sigma} \geq \frac{1}{\sqrt{2}}$ (see Fig.3.3 for the pseudo code of the algorithm). This proves that the approximation ratio is $1/\sqrt{2}$. On the other hand, if the splitting condition is not satisfied, then the partition is not performed and the subset is one of the clusters. The middle point of the rectangle diagonal is the Pseudo-Center of the cluster (see Fig. 3.2 for the pseudocode).

```
2D_DIAGONAL($S$, $Q$, Pseudo-Center)

Let $P_{X_m} \in S | P_{X_m}.x \leq P_i.x \ \forall P_i \in S$;
Let $P_{X_M} \in S | P_{X_M}.x \geq P_i.x \ \forall P_i \in S$;
Let $P_{Y_m} \in S | P_{Y_m}.y \leq P_i.y \ \forall P_i \in S$;
Let $P_{Y_M} \in S | P_{Y_M}.y \geq P_i.y \ \forall P_i \in S$;
$Q \longleftarrow$ FIND_ANTIPOLE($\{P_{X_M}, P_{X_m}, P_{Y_M}, P_{Y_m}\}$);
Pseudo-Center $\longleftarrow$ $((P_{X_m}.x + P_{X_M}.x)/2$
                  $, (P_{Y_m}.y + P_{Y_M}.y)/2)$;
return $\sqrt{(P_{X_m}.x - P_{X_M}.x)^2 + (P_{Y_m}.y - P_{Y_M}.y)^2}$;
END 2D_DIAGONAL
```

Figure 3.3: The algorithm to find the diagonal, the Pseudo-Diameter and the Pseudo-Center.

A variant of the above algorithm is the *bounded cluster size* version of the

Antipole Clustering. Here the splits occur also when the cardinality of the subset is larger than the size threshold k. This helps to satisfy throughput constraints (the condition to check the size of the cluster can be inserted between lines 3 and 4 in the pseudocode of Fig. 3.2).

In order to obtain an exponentially arbitrary low approximation ratio δ of the real diameter a bisecting algorithm is provided (see the pseudocode in Fig. 3.4). Performing a $\pi/4$ rotation of the Cartesian coordinates implies a bisection of the axes. Compute then, the maximum and minimum coordinate points for such two new axes, this yields a set of 8 points. Let (A, B) be the diameter of this set. Plainly, $\mathit{dist}(A, B)/\cos\frac{\pi}{8} > \mathit{Diameter}$ (Fig. 3.5 (b)). By iterating this bisection process d times we have $\mathit{dist}(A, B)/\cos\frac{\pi}{2^{d+2}} > \mathit{Diameter}$. Therefore, the approximation ratio is

$$\delta = \frac{|\mathit{Diameter} - \mathit{Pseudo\_Diameter}|}{\mathit{Diameter}} \leq \left|1 - \cos\frac{\pi}{2^{d+2}}\right| .$$

This yields the following Lemma:

Lemma 3.1.1. *Let* S *be a set of points in the plane and let* $0 < \delta \leq 1 - 1/\sqrt{2}$. *Then a call to* APPROX_DIAGONAL(S, δ) *returns an Antipole pair* (A, B) *(*$\mathit{Pseudo\_Diameter}$*) which approximates the diameter with an error bounded by* δ. ■

```
APPROX_DIAGONAL($S, \delta$)
Let $BBox = \{P_{X_m}, P_{X_M}, P_{Y_m}, P_{Y_M}\}$ be the minimum bounding box of $S$;
$V \leftarrow \{\{S\}\}$;
for $i = 1$ to $\lceil \frac{\pi}{4 \times \arccos(1-\delta)} - 1 \rceil$ do
    $V' = \mathrm{ROTATE\_SET}\left(V, \frac{\pi}{2^{i+1}}\right)$;
    Let $BBox_{\frac{\pi}{2^{i+1}}} = \{P_{X_m}, P_{X_M}, P_{Y_m}, P_{Y_M}\}$
    be the minimum bounding box of the rotated sets in $V'$;
    $V$ = Set catalog of $V'$;
    $BBox = BBox \cup BBox_i$;
end for
return FIND_ANTIPOLE($BBox$);
```

Figure 3.4: Algorithm for the Pseudo-Diameter Computation.

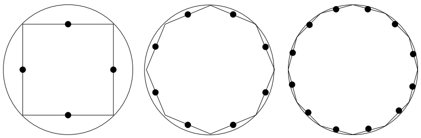

(a) (b) (c)

Figure 3.5: The worst cases in the first three iterations of algorithm in Fig.3.4.

3.2 Distributed Antipole Clustering and Base Stations Repositioning in Absence of Obstacles

Following [19], let n be the number of base stations and m be the cardinality of a set S of hosts. The algorithm partitions the hosts in disjoint subsets. Base stations occupy the centers of such sets. Suppose that the backbone has a specific interconnection topology such as a Euclidean Minimum Spanning Tree (EMST). Stations communicate through messages and each message is correctly transmitted to a unique successor which is its father in the EMST. There is one designated final base station f called *root* (not necessarily always the same) aggregating the results of all local computations.

Initially, the root f contains all the base stations coordinates, c_i^{old}, before the motion. After the motion, reclustering could be necessary. In such a case, the following steps of Distributed Antipole Clustering (DAC) procedure produces a set of new clusters, C_i^{new}, piecewise distributed among the base stations and whose centers, c_i^{new}, are all stored in the root f.

1. **Grouping**: Each base station j sends an HELLO message to hosts in order to build the group of nodes that it can hear.
2. **Local enclosing box computation**: Each base station j in the leaf in the EMST, computes the enclosing box $\{P^j_{X_m}, P^j_{X_M}, P^j_{Y_m}, P^j_{Y_M}\}$ of its local set of hosts S^j and passes such a bounding box to its father through a message of fixed length (see Figure 3.6 (a)).
3. **Global enclosing box computation**: Each internal node in the EMST receives enclosing boxes from its children and merges them with its local enclosing box. This merging results in the enclosing box of the enclosing boxes. This new enclosing box will be recursively propagated through the tree by using a fixed length message. At the end of this process the root station f will store the global enclosing box $\{P_{X_m}, P_{X_M}, P_{Y_m}, P_{Y_M}\}$.
4. **New cluster and center computation**: If the semi-diagonal of the global enclosing is smaller than the cluster radius threshold (communication radius) the root f sends a termination message to all stations (see Figure 3.6 (b)). Then, a base station will occupy the new center, c_i^{new}, of such an enclosing box.

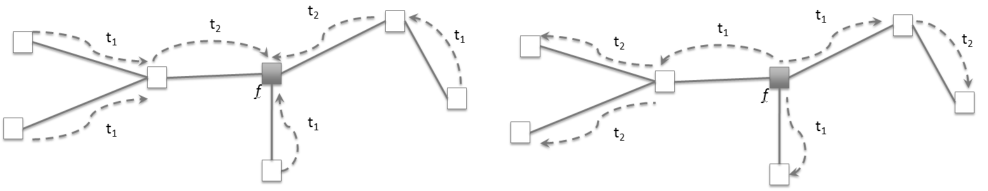
5. **Cluster splitting phase**: If the previous step does not apply, the root f broadcasts the farthest pair (global Antipole (A, B)) of the global enclosing box (see Figure 3.6 (b)). Each station, after receiving the global Antipole (A, B), splits its local data according to the distance from A and B. The algorithm proceeds recursively until splittings no longer occur.

6. **Termination**: When the splitting procedure is completed, each base station, j, contains the local portion, $C_i^{new,j}$, of hosts for every cluster, C_i^{new}. The root f contains all the cluster centers c_i^{new}.

7. **Base stations repositioning in absence of obstacles and backbone reconfiguration**: The root f, knowing the old, c_i^{old}, and the new centers, c_j^{new}, computes, through a bipartite matching algorithm (Closest Match [20] which tries to minimize the total distance covered by the base stations motion), the new position of each base station. Next, it calculates the new backbone EMST and the new root node. Such a node is the center of the EMST balancing the tree. Such a choice allows to minimize the number of hops needed to reach every base station during the clustering phase. Finally, the root f communicates to each station its new destination and the base stations reposition, accordingly (see Figure 3.6 (b)).

Concerning the last paragraph listed above, the computation of the new root node takes into account what follows [8]:

Lemma 3.2.1. *For any tree, the center of a tree consists of at most two adjacent vertices.*



(a) (b)

Figure 3.6: (a) Propagation of Local Enclosing Box Messages (Step 2 of the distributed Antipole Clustering algorithm). At time t_1 each leaf computes the local enclosing box and sends it to its father. Each internal node in the EMST merges the received enclosing boxes with its own. At time t_2, the internal nodes propagate the new enclosing boxes to the root station f which will compute and store the global enclosing box. (b) Propagation of messages from the root station f to the base stations through the EMST (Steps 4, 5, 7 of the distributed Antipole Clustering algorithm algorithm).

The proof of the lemma is trivial and gives a way to compute the center of the tree just by recursively deleting the leaves.

The above lemma implies that, in principle, each base station can be designated as a root in the Euclidean spanning tree. The root does not need extra energy because the additional effort required is only the computation of new base stations destinations. This allows to randomly change the root base station according to the current spanning tree.

The DAC algorithm is periodically executed by the base stations upon a request from the root. For each time interval Δt (20 seconds in our simulation), the root f collects the percentage of covered hosts h_t obtained from the sum of covered hosts given by each base station. Then, the root performs a *counter based procedure* (see Figure 3.7) to decide whether a clustering is needed. A global variable *counter*, initially set to zero, is increased when h_t falls below a certain threshold. If *counter* exceeds a prefixed value

$MAXCOUNTER$ (3 in our simulation), a new clustering starts and the *counter* is reset. From figure 3.7 we can see that, if the percentage of covered hosts falls below 95% then a new clustering will start. If the percentage of covered hosts falls below 98% a new clustering, at next round in the worst case will start (but it can start also instantly if *counter* reaches or exceeds $MAXCOUNTER$). Otherwise, counter will be increased by one. This ensure that, in the best case (i.e. in each round the percentage of covered hosts is above 98%), the clustering procedure will start anyway after three rounds (60 seconds in our simulation) allowing a better organization of the topology.

```
$checkClusteringCondition(h_t, counter)$

if $h_t \leq 95\%$ then $counter += 3$;
else if $h_t \leq 98\%$ then $counter += 2$;
else $counter++$;
if $counter \geq MAXCOUNTER$ then
    $counter = 0$;
    return $true$;
end if
return $false$;
END $checkClusteringCondition$.
```

Figure 3.7: Clustering Condition Check. At each round a counter based check is performed. If the procedure returns true, a new clustering starts.

Since the Antipole Clustering does not guarantee that the number of clusters' centroids is equal to the number of base stations (i.e. the number of destinations may differ from the number of base stations), the root f proceeds accordingly:

- If the number of centroids is less than the number of base stations, some of such base stations (the furthest from the centroids) stay idle but still used for communications. This is achieved without any additional cost by the matching algorithm which will match only the correct number of base stations leaving unmatched the exceeding ones.

- When the number of clusters exceed the number of base stations, clusters are merged hierarchically (i.e. by using neighbor-join [6]) until the right number of clusters is reached.

3.3 Clustering and Base Stations Repositioning in Presence of Obstacles

In a real situation base stations move in a relatively open space in presence of obstacles (i.e. hills, rivers, buildings, urban roads etc). Without loss of generality, in our proposed algorithm, we approximate obstacles by their minimum bounding rectangle containing it and whose sides are parallel to the coordinate axes. With the presence of obstacles we extend the protocol described in Section 3.2 to capture the following scenarios.

Obstacles may obstruct the signal propagation. Nodes that are obstructed by obstacles may still be able to communicate, but the received signal has lower power level. Therefore, we introduce a signal attenuation factor, A_t ranging from 0 to 1 (A_t=0 means that obstacles do not influence the transmission, A_t=1 indicates that obstacles completely obstruct the signal propagation).

Cluster centers may fall within an obstacle. In order to ensure that target positions are always placed outside obstacles, we modified the step 4 in Section 3.2 as follows. If the center of the global enclosing box c_i^{new} falls inside an obstacle the root f moves it perpendicularly to the closest side of the obstacle. Then, if the furthest vertex of the cluster enclosing box is greater than the cluster radius threshold (communication radius), the root f splits the cluster (go to step 5 in the preceding Section). Otherwise it sends the termination message with the new cluster center.

Obstacles may obstruct the paths of base stations. In this case, we choose to minimize the parallel motion time needed to complete the repositioning (i.e. we minimize the maximum distance). For this purpose we adpated the Bottleneck Matching algorithm [1] on Visibility Graphs [9, 47]. On the other hand, the Minimum Weight Bipartite Matching algorithm [2] can be used to minimize the total distance covered by the base stations. More precisely, we perform base station repositioning in the following way. Let $G(V, E)$ be the visibility graph [47] constructed on the set O of obstacles vertices, let $c^{old} = \{c_1^{old}, c_2^{old}, \cdots, c_n^{old}\}$ be the current base stations positions, and let $c^{new} = \{c_1^{new}, c_2^{new}, \cdots, c_n^{new}\}$ be the base stations destinations.

Edges are weighted, w, with the Euclidean distance. Distances between pair of nodes are given by the Dijkstra algorithm [13]. A matching M, called *bottleneck match*, is a collection of n paths $\pi_1, \pi_2, \cdots, \pi_n$ connecting sources, c^{old}, to destinations, c^{new}, in a one-to-one fashion. A bottleneck match minimizes the parallel time function: $PT_M(c^{old}, c^{new}) = \max_{1 \leq i \leq n} \sum_{e \in \pi_i} w(e)$[1]. In order to simplify presentation, we assume the absence of collisions among base stations during the motion (see [9] for collisions avoidance). In what follows we give a sketch of the steps of the bottleneck matching.

1. **Initialization**: Initialize D to be an ordered set of all shortest path

---

[1] This problem with the condition stating that two paths cannot collide is NP-complete [15]. By relaxing such a condition (disjointness of the paths) the problem becomes polynomial.

distances among pairs of sources and destinations (c_i^{old}, c_j^{new}).

2. **Iteration**: Let m be the median of D, search for a match M such that $PT_M(c^{old}, c^{new}) \leq m$.
    - **Match found**: If a match M of cardinality n exists, then try to find a match recursively on the first half D.
    - **Match not found**: try to find a match recursively on the second half of D.
3. **Termination**: The search stops when current D is empty. The last successful match M represents the output of the algorithm.

The key step of the bottleneck matching algorithm is the construction of the partial matching during the above binary search strategy. Starting from a match M, construct incrementally a match of cardinality $|M|+1$. The notions of alternating and augmenting walks are introduced. A current match M in the visibility graph $G(V, E)$ is a set of paths starting from a node of c^{old} and ending in a node on c^{new}. We label those nodes *matched* and the remaining ones *exposed*. Notice that internal nodes of a path in M, if any, can be only vertexes of obstacles.

A path $\pi = (v_1 \rightsquigarrow v_2 \rightsquigarrow v_3 \rightsquigarrow \ldots \rightsquigarrow v_{2t})$ is called an *alternating walk* if:

1. v_1 is an exposed vertex of c^{old} and v_{2t} is in c^{new};
2. all the paths $(v_{2i} \rightsquigarrow v_{2i+1}) \in M$, with $i = 1, \cdots, t-1$, and $(v_{2i-1} \rightsquigarrow v_{2i})$, with $i = 2, \cdots, t$, are paths in G which are not in M and their length is less than m.

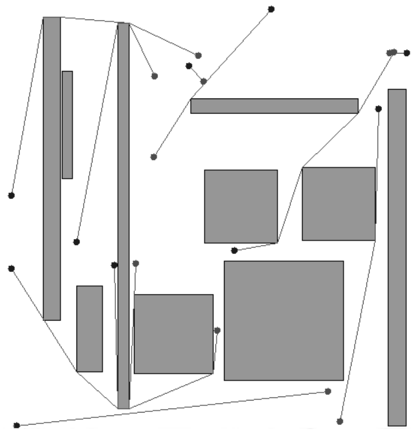


Figure 3.8: An example of Bottleneck matching in the presence of obstacles. Each obstacle is approximates by its minimum bounding box.

An alternating walk is called an *augmenting walk* if v_{2t} is exposed. If π is an augmenting path then $M' = (M \backslash \{\cup_{i=1,\ldots t-1}(v_{2i} \rightsquigarrow v_{2i+1})\}) \cup \{\cup_{i=2,\ldots t}(v_{2i-1} \rightsquigarrow v_{2i})\}$ (obtained by removing from M the paths of π belonging to it and adding the remaining paths of π) is a match and $|M'| = 1 + |M|$.

Complexity Analysis. The complexity of the proposed algorithm is higher than the corresponding Euclidean version since basic computational geometry data structures are not available on graphs. Given h obstacles and n base stations, each obstacle bounding rectangular box has 4 vertices, and for each base station there is a starting and ending position, than the total number of nodes in the visibility graph will be $2n+4h$. If the visibility graph has k edges, then by extending the arguments of [1] the complexity of the proposed algorithm turns out to be $O\,(n \times ((2n+4h)\ \log(2n+4h)+k)+n^2)$.

Chapter 4

Throughput Optimization and Network Maintenance

Communication among hosts in our model is guaranteed by a two level hierarchy architecture (see Figure 4.1). In the first level, a host who wants to communicate sends a message to the joint base station. In the second level, by using the standard AODV, base stations identify the route to reach the destination host. The communication is guided by the AODV protocol. Therefore, the backbone used for communication differs from the one adopted during the clustering (i.e. the EMST).

Since hosts frequently move, to maintain up to date the network and to optimize the base stations load, hosts are periodically assigned according to a probabilistic throughput function. The throughput combines the quality of the signal (influenced by the presence of obstacles and in general inversely proportional to the distance host - base station) and the potential load of the base stations due to the number of hosts in their neighborhood. Each host

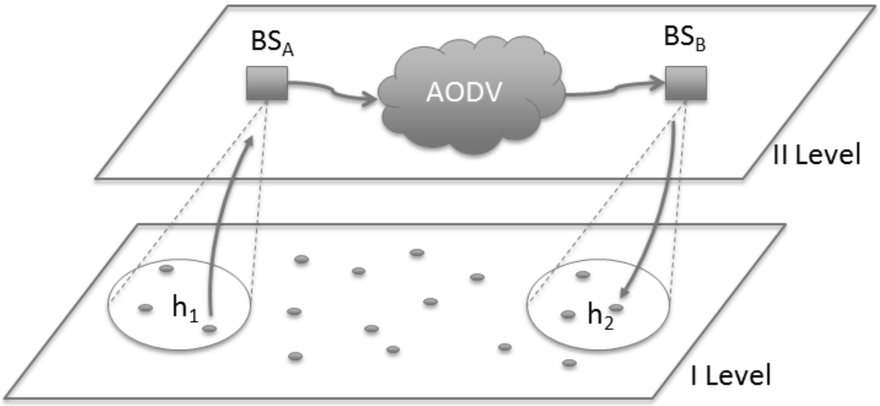


Figure 4.1: The two level hierarchy architecture

computes such a function yielding a score that identifies the best base station to join with. Since we use a single communication channel (802.11x protocol) the throughput achieved by a host transmitting to its assigned base station is influenced by the totality of the hosts in the same neighborhood and not only by those joint with the base station [48]. For this reason, models based on channels separation [45, 44] cannot be applied.

More precisely, hosts assignment is achieved in the following way. Similarly to the AODV, each host h_j periodically (i.e. every 3 seconds in our simulation) receives an HELLO message with a certain quality (inversely proportional to the distance) from the base stations which it can hear. Each HELLO message, sent by the base station, BS_i, contains n_i, the number of hosts reachable by BS_i (i.e. number of hosts assigned to such a base station plus the hosts in its neighborhood which answered to the previous HELLO message). Suppose that the host h_j can hear k base stations, each of them with quality $p_{ij} \propto 1/d(h_j, BS_i)$ $(i = 1, ..., k)$. The probability that the host

h_j will join the i-th base station is defined as follows:

$$TP(p_{ij}, n_i) = \frac{\frac{p_{ij}}{(n_i+1)^\alpha}}{\sum_{l=1}^{k} \frac{p_{lj}}{(n_l+1)^\alpha}}$$

It is easy to see that fixed a base station, $TP()$ is directly proportional to the power of the HELLO message sent and it is inversely proportional to the number of hosts in the base station neighborhood. The denominator is used to normalize $TP()$ in the $[0, 1]$ interval.

In the experimental section we show that $TP()$ allows to increase the percentage of delivered messages with respect to an assignment based only on distances.

Although the host assignment based on the $TP()$ function works well in practice, it is a heuristic assignment and it can not guarantee the best allocation.

Finally, notice that, the above network maintenance is also performed during base station repositioning (see Figure 4.2). This ensures hosts communications and avoids network stalls.

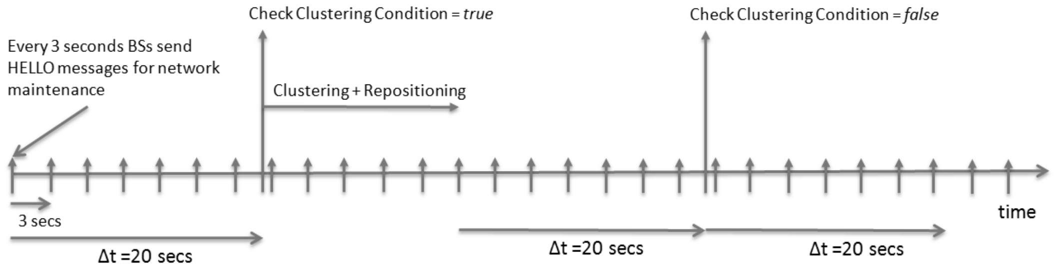


Figure 4.2: Network infrastructure temporal maintenance. After the first 20 seconds a clustering is needed and thus performed. On the contrary, the second check condition results false and the clustering does not start. In order to ensure network maintenance, base stations send HELLO messages also during clustering and repositioning.

Chapter 5

Introduction to NS2

5.1 NS2 Network Simulator

Network Simulator (Version 2), widely known as NS2, is a powerful event-driven simulation tool that has been widely used in studying the dynamic nature of communication networks. Through NS2, wired and wireless networks functions and protocols (e.g., routing algorithms, TCP, UDP) can be simulated and analyzed. In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviors. Due to its modular nature, NS2 has gained constant popularity in the networking research community. Thanks to substantial contributions from several research group, extensions and revisions have marked the growing maturity of the tool. NS2 is an evolution of Network Simulator 1 developed through a joint project between University of California and Cornell

University. Since 1995 the Defense Advanced Research Projects Agency (DARPA) supported development of NS through the Virtual InterNetwork Testbed (VINT) project. Currently the National Science Foundation (NSF) has joined the development project. Last but not the least, there is a wide community of researchers and developers who constantly work on NS2. Figure 5.1 shows the basic architecture of NS2. NS2 provides users with an executable command `ns` which takes as input the name of a Tcl simulation scripting file. In most cases, a simulation trace file is created, and is used to plot graphs and/or to create animations. NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a back-end) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events (i.e., a front-end). The C++ and the OTcl are linked together using TclCL [18]. Mapped to a C++ object. The variables in the OTcl domains are sometimes referred as handles. Con-

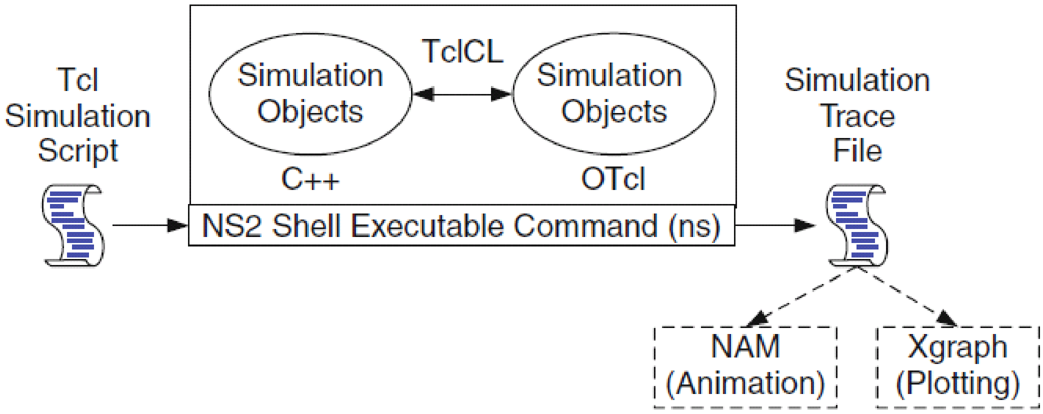


Figure 5.1: Basic architecture of NS.

ceptually, a handle (e.g., n as a Node handle) is just a string (e.g., `_o10`) in the OTcl domain, and does not contain any functionality. Instead, the functionality (e.g., receiving a packet) is defined in the mapped C++ object (e.g., of class Connector). In the OTcl domain, a handle acts as a front-end which interacts with users and other OTcl objects. It may defines its own procedures and variables to facilitate the interaction. The member procedures and variables in the OTcl domain are called instance procedures (instprocs) and instance variables (instvars), respectively. NS2 provides a large number of built-in C++ objects. It is advisable to use these C++ objects to set up a simulation using a Tcl simulation script. However, users could need to develop their own C++ objects, and use a OTcl configuration interface to put together these objects.

After simulation, NS2 outputs either text-based or animation-based simulation results. To inspect these results graphically and interactively, tools such as NAM (Network AniMator) can be used. 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. NAM however is not able to show items such as obstacles. To work around this limitation we used fake wired networks (see figure 5.2), irrelevant to the simulations.

To analyze a particular behavior of the network, we can extract subsets of text-based data and transform them into a more conceivable presentation. Simulation data is acquired from trace-files which NS2 generates. Trace-files can either filtered at runtime by using a script or can be written directly to disk. Both methods present drawbacks. Filtering at runtime could discard

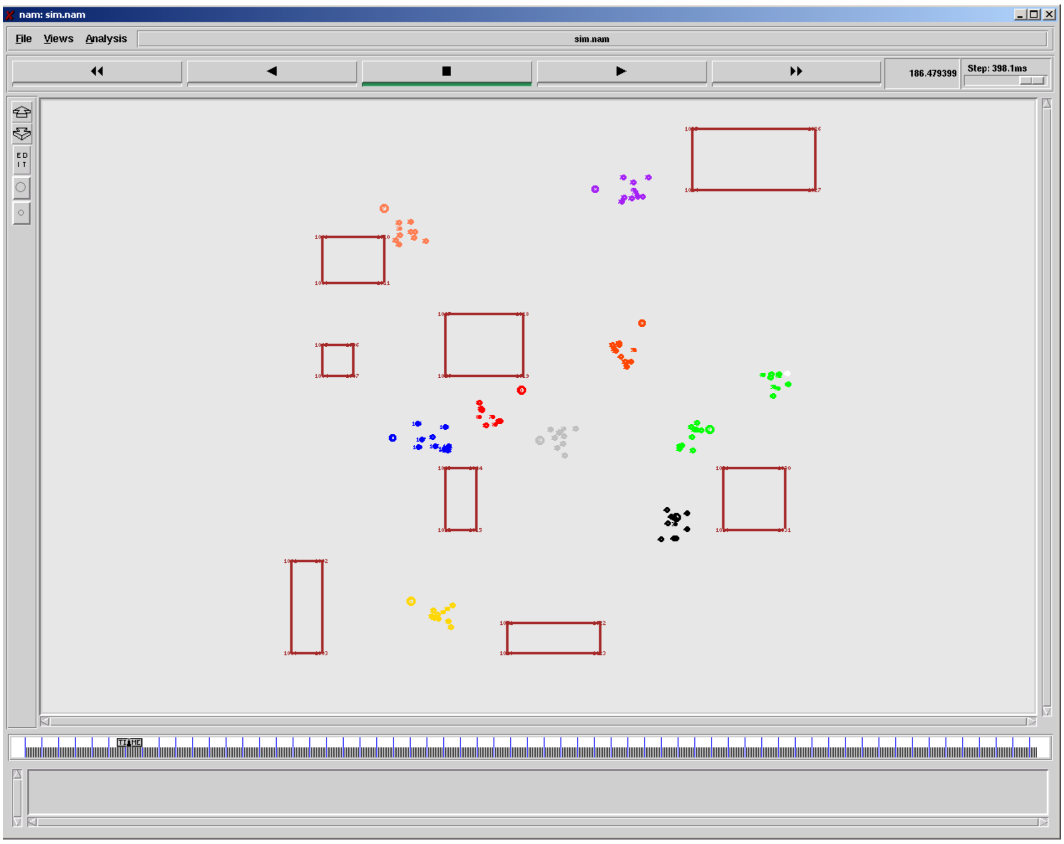


Figure 5.2: NAM example.

relevant data, writing to disk may take too much time and trace-files from simulations could need many Gigabytes of space. Our model of simulation requires a OTcl script for network configuration, a mobility pattern describing nodes movement, a traffic pattern describing data traffic and a file describing coordinates obstacles.

5.2 Layers into NS2

A computer network is a complex system. To facilitate design and flexible implementation of such a system, the concept of *layering* is introduced. Using a layered structure, the functionalities of a computer network can be organized as a stack of layers. There is a peer-to-peer relationship (or virtual link) between the corresponding layers in two communicating nodes. However, actual data flow occurs in a vertical fashion from the highest layer to the lowest layer in a node, and then through the physical link to reach the lowest layer at the other node, and then following upwards to reach the highest layer in the stack. Each layer represents a well-defined and specific part of the system and provides certain **services** to the above layer. Accessible (by the upper layers) through so-called **interfaces**, these services usually define *what* should be done in terms of network operations or primitives, but does not specifically define *how* such things are implemented. The details of how a service is implemented is defined in a so-called **protocol**. For example, a source node transmitter can use at the physical layer a specific protocol (e.g., a data encoding scheme) to transmit data to a receiver, which should be able to decode the received information based on the protocol rules. The beauty of this layering concept is the layer independency. That is, a change in a protocol of a certain layer does not affect the rest of the system as long as the interfaces remain unchanged. Here, we highlight the words services, protocol, and interface to emphasize that it is the interaction among these components that makes up the layering concept. As shown in figure 5.3 NS2 works with layers, where each layer is a module or a set of modules written

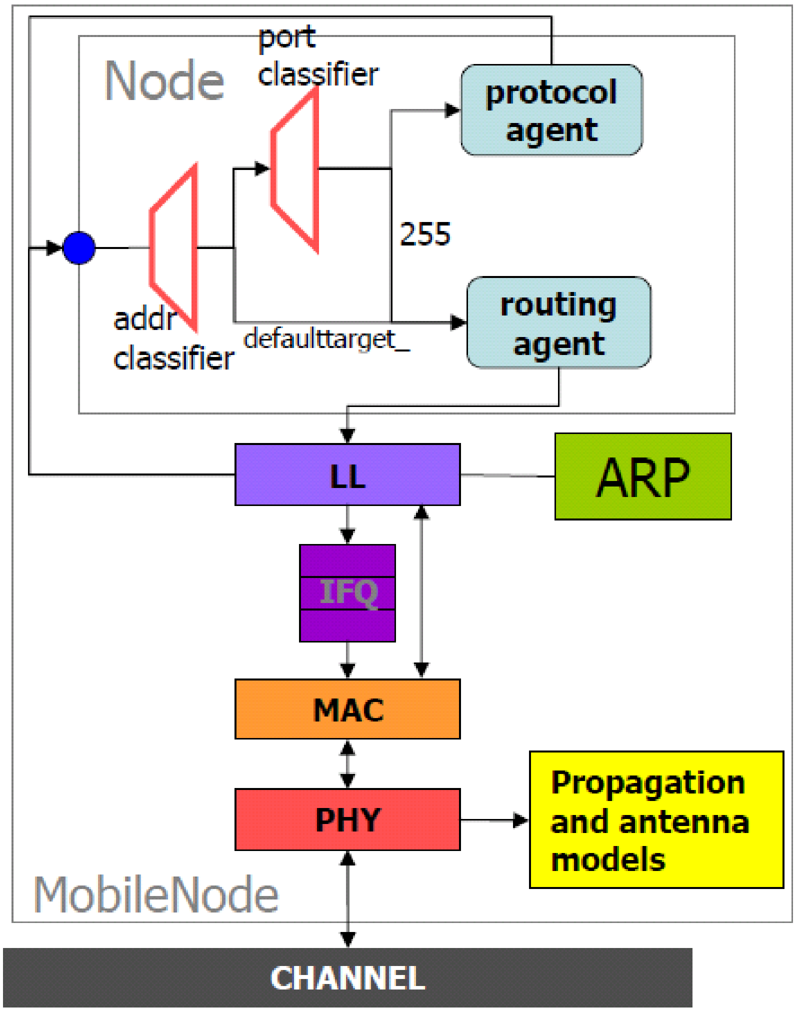


Figure 5.3: Layers into NS2.

in C++.

5.2.1 Physical Layer

Physical layer (often termed PHY) is referring to network hardware, physical cabling or a wireless electromagnetic connection. It provides an electrical, mechanical, collision control and procedural interface to the transmission medium. The Physical Layer defines the means of transmitting raw bits rather than logical data packets over a physical link connecting network nodes. The bit stream may be grouped into code words or symbols and converted to a physical signal that is transmitted over a hardware transmission medium. The physical layer is the very simplest, defining only exactly what a bit is: in other words how to transmit a one or a zero.

The physical layer of wireless simulation in NS2 contains an important module: the Signal Propagation Model. It determines whether two nodes can communicate: by distance, signal strength and by other additional variables.

5.2.2 The Signal Propagation Model

One of the primary limitations of the performance of wireless networks is the significant attenuation and interference experienced by the radio signal as it propagates from the sending node to the receiving node. In a setting with obstacles, the signal may reach the receiver via non-line-ofsight propagation mechanisms, such as reflection, diffraction and scattering. This effect of multi-path propagation results in a drop in the Signal-to-Noise Ratio (SNR) of the received signal. The free-space fading models are not suitable to calculate the attenuation undergone by the signal being received. Additionally, the fluctuations of the signal levels are log-normally distributed about a mean value and the changes in the signal levels are insignificant over short periods of time, leading to a phenomena called long-term fading. In our simulations, we use either the Two-Ray Pathloss Model that accommodates the reflections of the signals off the surface of the ground, in addition to the direct path signals from the source transceiver to the destination transceiver; or the Friis' Free Space Equation [42] which considers only a single path of propagation. The equation is defined as follows:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2}$$

where P_r is the Power received, P_t is the power transmitted, G_t and G_r are the antenna gain of the transmitting and receiving antennas, respectively, λ is the wavelength of the radio signal and d is the distance. The choice of these models depends on the value of the cross-over distance, d_c, where d_c is the distance such that when $d = d_c$, the received power predicted by the two-ray ground model is equal the one predicted by the Friis equation. Two case are possible:

- when $d < d_c$, we use the Friis' Equation since the power attenuated is inversely proportional to d^4;
- when $d \geq d_c$, we use the Two Ray Pathloss Model since the power attenuated is inversely proportional to d^2;

In detail $d_c = \frac{4\pi h_t h_r}{\lambda}$, where h_t and h_r are the antenna heights of the transmitter and the receiver, respectively.

The Two Ray Pathloss Model equation is defined as:

$$P_r = \frac{P_t G_t G_r (h_t h_r)^2}{d^4}$$

In the real world a radio signal transmitted between a pair of nodes undergoes fading, attenuation, scattering, diffraction, reflection, multipath propagation, etc. In this thesis we use empirical results to simulate these effects. Hence, there is a possibility that two nodes that are obstructed by an object or any other natural obstacle may still be able to communicate, but the signals that are received have a lower power level. Before a simulation begins, we can specify the penetration characteristics of a obstacle. There are two extreme cases for this specification. In the case of a perfect conductor, a radio wave that is incident to the material is completely attenuated (i.e., Attenuation $A_t \rightarrow \infty$). On the other hand, an obstacle can be specified such that it obstructs only movement, and not the propagation of radio signals, e.g, a river. In this case, the radio wave does not fade due to the obstacle (i.e., $A_t = \mathit{0}$). Hence, $\mathit{0} \leq A_t \leq \infty$. As mentioned in section 3.3, for simplicity our A_t is a value between 0 and 1.

A text file, named Obstacles.txt, is used to describe the obstacles in the scenario of our simulations. For simplicity, each obstacle is represented by a rectangle (as mentioned in section 3.3), as an ordered sequence of its four vertices whose sides are parallel to the coordinate axes, and its A_t value. Starting from the standard TwoRayGround module (written in C++) of NS2 to simulate a signal propagation with Two-Ray Pathloss Model, we have developed TwoRayGroundwObs able to manage obstacles in a wireless simulation.

Chapter 6

Implementation and Experimental Analysis

6.1 Implementation

In this section, we present an implementation of our model as an extension of the widely used Ad-hoc On-Demand Distance Vector [40] protocol (AODV). Such a modeling is suitable for our purposes since AODV is based on HELLO messages.

In order to simulate the presence of a base stations backbone, we adapted the AODV by introducing a two level hierarchical network organization called Two-Level-Hierarchy-AODV (TLH-AODV). As mentioned in section 5.1, NS2 is an object oriented simulator written in OTcl and C++ languages. While OTcl acts as the front-end (i.e., user interface), C++ acts as the back-end running the actual simulation. As can be seen from Fig. 6.1, class hierarchies of both languages can be either standalone or linked together using an OTcl/C++ interface called TclCL [18] . There are two types of classes in each domain. The first type includes classes which are linked between the C++ and OTcl domains. In the literature, these OTcl and C++ class hierarchies are referred to as *the interpreted hierarchy* and *the compiled hierarchy*, respectively. The second type includes OTcl and C++ classes which are not linked together. These classes are neither a part of the interpreted hierarchy nor a part of compiled hierarchy.

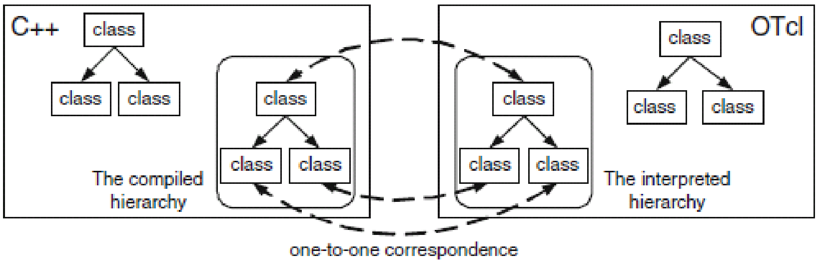


Figure 6.1: Two language structure of NS2. Class hierarchies in both the languages may be standalone or linked together. OTcl and C++ class hierarchies which are linked together are called the interpreted hierarchy and the compiled hierarchy, respectively.

As discussed above, TwoRayGroundwObs implements the signal propagation Model in the presence of obstacles (see Sections 5.2.2 and 3.3) and TLH – AODV implements the proposed protocol (see Chapters 3 and 4). In order to generate mobility patterns, we did not use the built-in program ”setdest” of NS2 because it implements only Random Waypoint in absence of obstacles. For this reason we rewrote completely the procedure for generating mobility patterns for both RWP and RPGM in the presence of obstacles.

As shown in Fig. 6.2, in order to take into account the new mobility models and algorithms, ad hoc parameters have been introduced in the simulation

```
# Define options
# Set variable           Value                           Comment

set val(chan)            Channel/WirelessChannel         ;# Channel type
set val(prop)            Propagation/TwoRayGroundwObs    ;# radio-propagation model
set val(netif)           Phy/WirelessPhy                 ;# network interface type
set val(mac)             Mac/802_11                      ;# MAC type
set val(ifq)             Queue/DropTail/PriQueue         ;# interface queue type
set val(ll)              LL                              ;# link layer type
set val(rp)              TLH_AODV                        ;# routing protocol
set val(x)               900                             ;# X dimension of topography
set val(y)               900                             ;# Y dimension of topography
set val(stop)            1800                            ;# time of simulation end
set val(emodel)          EnergyModel                     ;# Energy Model
set val(sigma)           200                             ;# SIGMA value in Antipole Clustering
set val(clustertime)     20                              ;# Time value beetwen two clustering
.......
```

Figure 6.2: The initial portion of environment definition, written in Tcl, of a our simulation.

script.

The simulations were done using NS2 v. 2.33 under a Pentium IV 2.8GHz with 1GB RAM with Linux operating system (kernel 2.6). Table 6.1, shows the parameters used during the simulation.

Since we assume a single communication channel we do not compare our model with those presented in [44, 45] because they are based on channel separation: hosts which are close but joint with different base stations do not interfere with each other. Furthermore, in [45] the problem solved is slightly different because the number of available base stations is not bounded (in our case the number of base station is fixed a-priori). Concerning [44], the protocol is an interesting theoretical result. Indeed, it results unfeasible because of its high computational complexity. In the presented scenarios we need efficient methods to frequently reassign hosts and move base stations.

For this reason, we decided to compare our model with two standard

Parameter	**Value**
Simulation area size	$900m \times 900m$
Host-bs transmission range	$150m$
Bs-bs transmission range	$400m$
MAC 802.11 data rate	$11Mb/s$
Wireless phy frequency	$2.4GHz$
Signal propagation model	*two ray ground* (adapted to deal with obstacles obstructing the communication)
Hosts walking speed	$0.5m/s$ - $1.8m/s$
Base stations speed	$15m/s(54Km/h)$
Simulation timing	1800 seconds
Number of hosts	100
Number of BS	9
CBR communication packets	2000 (between two randomly chosen hosts)
Clustering time check	20 seconds
MAXCOUNTER	3
α	0.25

Table 6.1: Simulation parameters.

wireless protocols such as AODV and DSR.

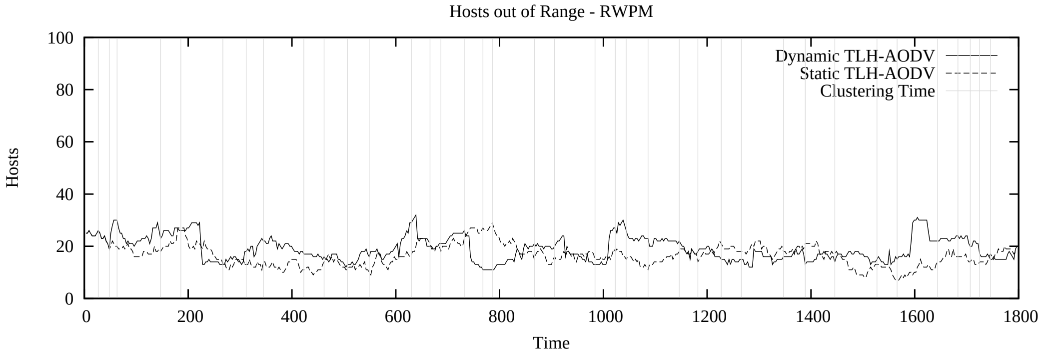
In order to highlight the strengths and the weakness of our model we performed several experiments.

We ran the simulation for 100 pairs of randomly chosen connections comparing the behavior of the protocol on two different mobility models: RWPM and RPGM. Hosts turn around obstacles when they obstruct their way and the protocol takes into account a signal attenuation factor. In the RWPM, nodes are initially randomly placed in the simulation area and move according to the model. In the RPGM, nodes are organized in groups (in our simulation we considered 10 groups) and each group moves randomly in the area. Both models take into account the obstacles randomly distributed in de simulation area. We analyzed the performances of two different versions of our protocol: the *Dynamic TLH-AODV* in which bases stations move and the *Static TLH-AODV* in which base stations are fixed and no clustering is performed. In the latter case, base stations are placed in the simulation area in order to maximize the space covered. We performed several tests to establish the α parameter of $TP()$ which has been set to 0.25 for all the experiments.

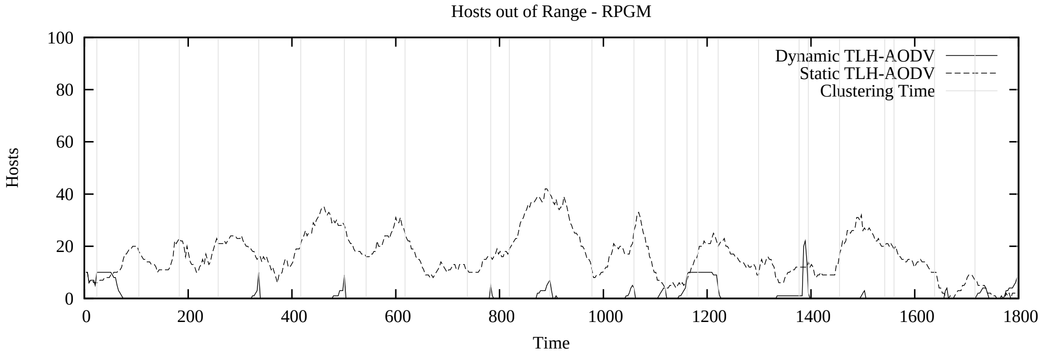
6.2 Experimental Analysis

In Fig. 6.3 we report the number of hosts out of range on both mobility models. Experiments show that with RWPM (Fig. 6.3 (a)) the number of lost hosts ranges from 10% to 30% with both versions of the protocol (static/dynamic). This poor and unstable coverage is essentially due to the uniform distribution and movement of mobile hosts. Moreover, we can observe that, when an accurate static base stations positioning occurs, the static protocol has the same performance of the dynamic one. On the other hand, using the more realistic RPGM (Fig. 6.3 (b)) the Dynamic TLH-AODV protocol guarantees 100% of coverage almost in all the simulation together with a lower number of reclustering.

In Fig. 6.4, we report the energy spent by the hosts. x axes represents the hosts and the ordinates contain the total energy consumed by each of them with respect to the number of delivered packets. Here, we compared our protocols (static and dynamic) with respect to AODV and DSR [31]. Using RWPM our models have the best behavior, whereas with RPGM they are comparable. In Fig. 6.5, we report the percentage of delivered packets using both RPGM and RWPM models. By using RPGM (Fig. 6.5 (a)), the dynamic TLH-AODV yields the best performances guaranteeing the highest percentage of delivered packages. By Increasing the traffic we observe that all the protocols converge to a similar behavior. This is due to the high load of the network. On the other hand with RWPM our protocol has a worse behavior than AODV and DSR ((Fig. 6.5 (B)). These results are completed with those presented in Figg. 6.6 and 6.7 in which we report the percentage



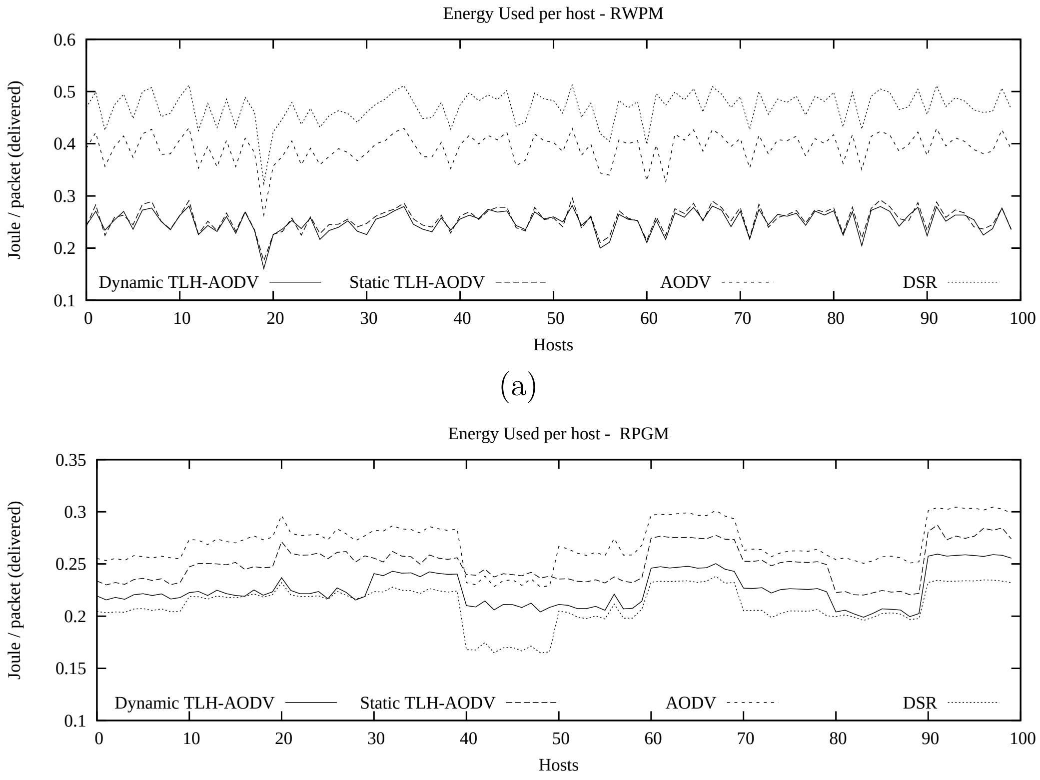
(a)



(b)

Figure 6.3: Number of hosts out of range for each mobility model. (a) Comparison of coverage using RWPM. (b) Comparison of coverage using RPGM. The vertical bars in both graphs depict the time in which the reclustering occurs. Clearly, the Dynamic TLH-AODV performs fewer reclustering than Static TLH-AODV.

of dropped packets (RTR IFQ, RTR NRTE and MAC). The RTR IFQ drops occur when a base station buffer has an overflow and the messages waiting to be sent are overwritten. The RTR NRTE drops occur when no route is available. The MAC layer drops occur because of higher traffic load [50]. With RPGM (Fig. 6.6), AODV has the largest percentage of IFQ and NRTE lost packets. This is due to the fact that hosts cannot establish bidirectional connections. On the other hand, the DSR protocol has static routing tables



(a)

(b)

Figure 6.4: Energy consumed with respect to the number of packages delivered. Four protocol have been compared: the standard AODV, the DSR, the Dynamic TLH-AODV, and the Static TLH-AODV. (a) Using RWPM, the AODV and the DSR have the worst ratio due to the number of lost packets. (b) Using RPGM, all protocols have a better behavior, however the Dynamic TLH-AODV and DSR show the best ratio.

and for this reason the largest percentage of lost packets is at MAC level. Results clearly show that using RPGM, a base station backbone infrastructure is needed to guarantee communication. By using RWPM (Fig. 6.7), TLH-AODV has the highest number of lost packets. This behavior is strictly related to the poor space coverage due to the uniform hosts distribution. In such a case several hosts are out of base station communication range.

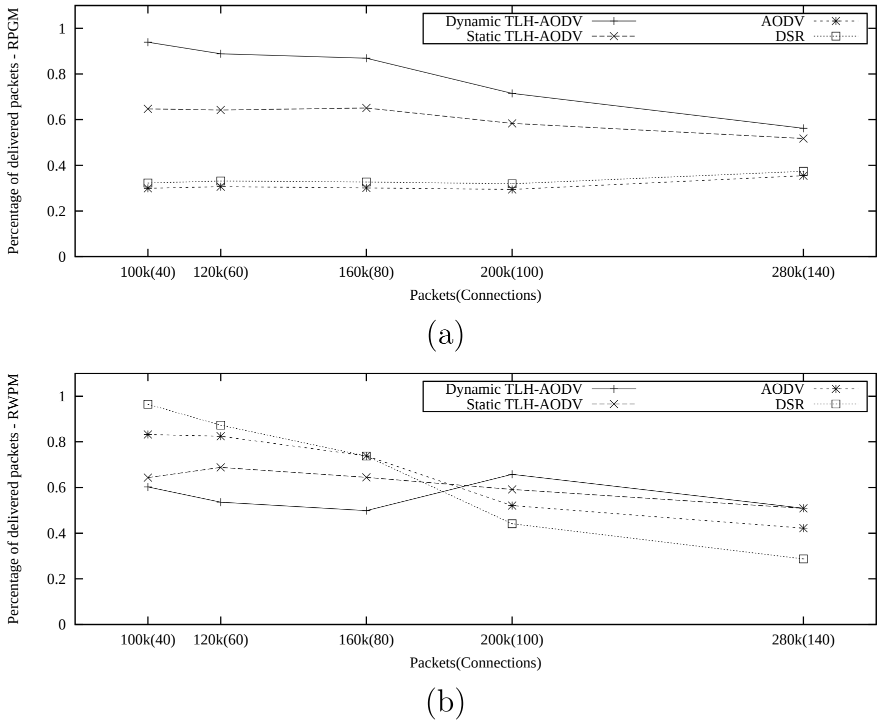


Figure 6.5: Percentage of delivered packets using both RPGM and RWPM models. The α is set to 0.25. (a) Using RPGM our dynamic TLH-AODV yields the best performances. (b) Using RWPM the AODV and DSR give the best percentage of delivered packets.

In Fig. 6.8, we show a comparison with RPGM of closeness based joining mode (based only on distances) with respect to the one based on the throughput function. The experiment highlights that the throughput function guarantees higher number of delivered packets. Also in this case, increasing the number of exchanged packets will cause a congestion of the network. This will reduce the different behavior of the two version of the protocol. We can conclude also that the proposed Dynamic TLH-AODV results the most robust since it drops less messages than the others especially when the number of exchanged messages increases.

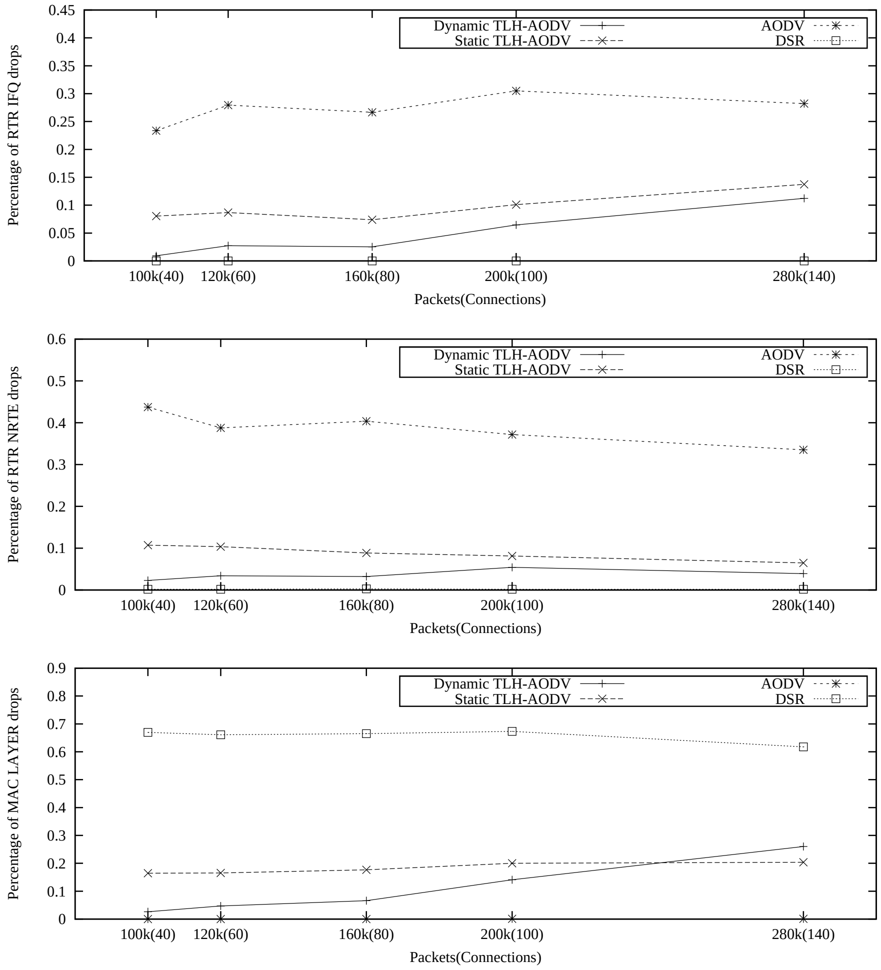


Figure 6.6: Percentage of lost packages (RTR IFQ, RTR NRTE, MAC) using RPGM. AODV has the largest percentage of RTR IFQ and RTR NRTE drops. This is clearly due to the fact that hosts cannot reach each other since they cannot establish bidirectional connections. DSR protocol has static routing tables, thus the the largest percentage of lost packages is at MAC level. Experiments clearly show that using the realistic RPGM to guarantee communication a base station backbone infrastructure is needed.

To consider a scenario in which the base stations are overloaded, we tested the model using a skewed traffic distribution. The hosts are partitioned in four groups generating traffic with this distribution: first group 4 hosts and

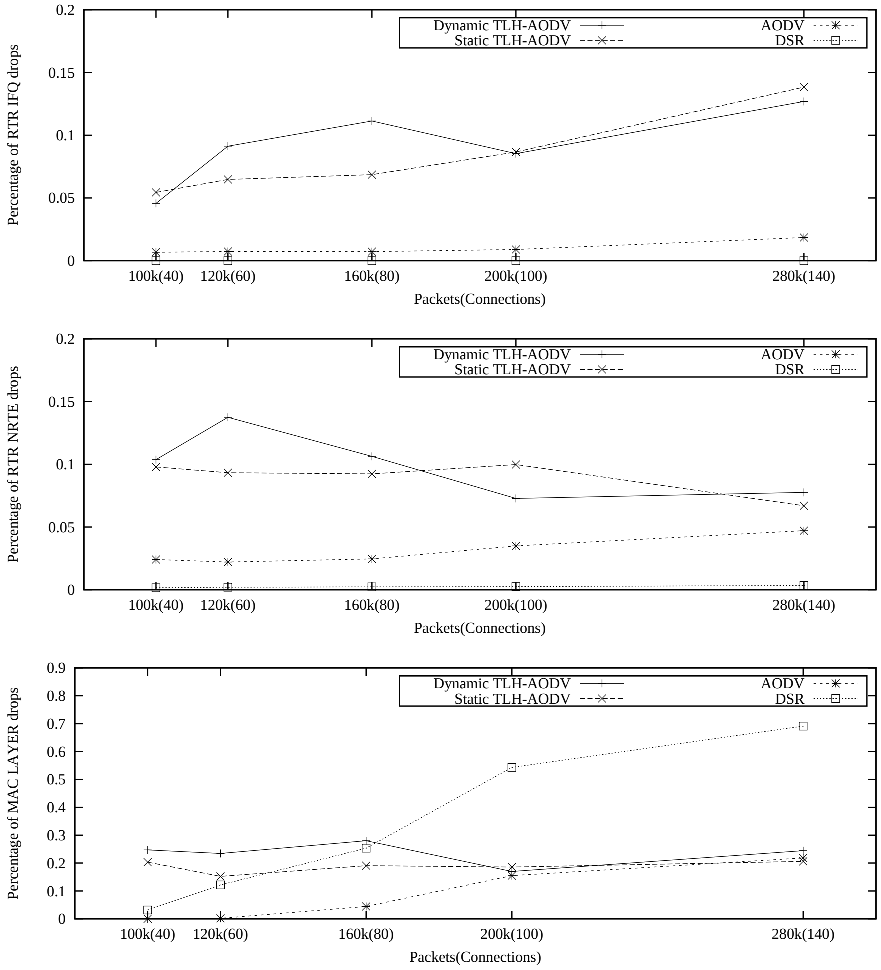


Figure 6.7: Percentage of lost packets (RTR IFQ, RTR NRTE, MAC) using RWPM. Here TLH-AODV has the highest number of lost packets. Since with RWPM hosts are uniformly distributed in the area the space coverage is not optimal. Thus, several hosts will be out of base station communication range.

32 connections, second group 8 hosts and 16 connections, third group 32 hosts and 8 connection, and fourth group 56 hosts and 4 connections. In Fig. 6.9, we present the percentage of delivered packages using an exponential distribution of exchanged messages with both RPGM (a) and RWPM

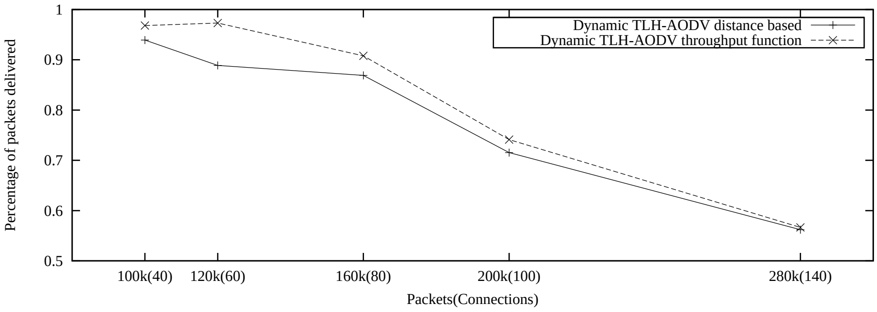
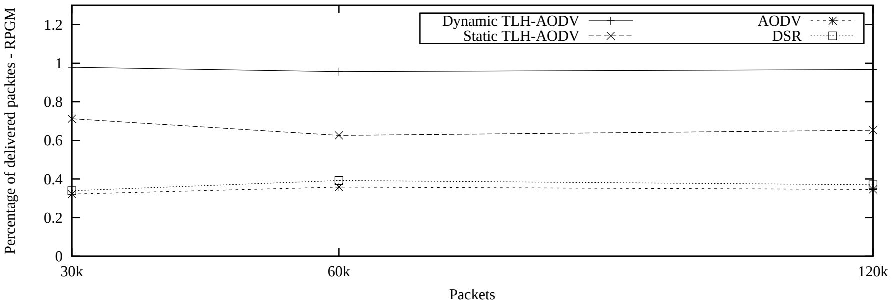
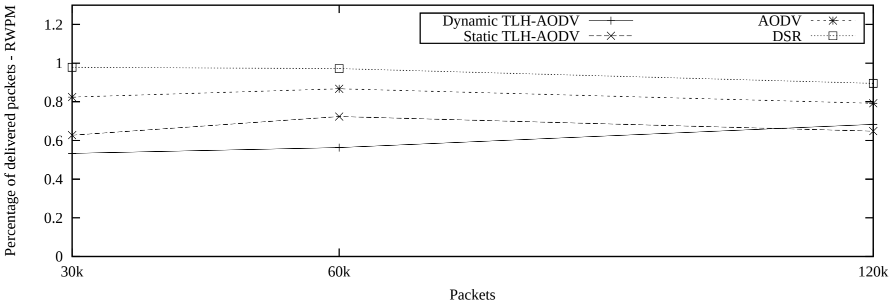


Figure 6.8: A comparison of closeness based joining mode with respect to the one based on the throughput function ($\alpha = 0.25$ and mobility model RPGM).

(b). The dynamic TLH-AODV has the best behavior using RPGM. On the other hand, with RWPM, DSR shows the best performances. The percentage of drop packets are similar to those reported in the previous experiments and for this reason are not presented. In Fig. 6.10, we analyzed the quality of the received signal of each host. In this case, the Dynamic TLH-AODV protocol guarantees a better signal for the hosts. This is due to the fact that base stations move according the hosts movements and are always placed in the clusters' centroids. In Figg. 6.11 and 6.12, we analyze the computational effort needed to cluster the hosts space using RPGM. In Figg. 6.11 (a) and (b), we report the time needed to cluster the hosts space varying the number of hosts and the number of base stations, respectively. In Figg. 6.12 (a) and (b), we report the number and type of messages sent during the clustering construction. In Table 6.2 we report the size and the description of each message. All the messages have a fixed length except for the "Set position" which is proportional to the number of base stations. Clustering scales linearly with respect to the number of base stations and its running time



(a)



(b)

Figure 6.9: Percentage of delivered packages using an exponential distribution of exchanged messages with both RPGM (a) and RWPM (b).

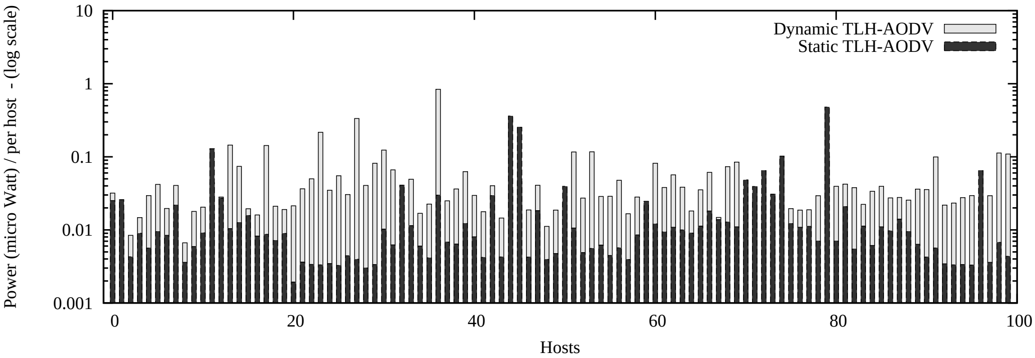


Figure 6.10: Average quality of the received signal for each host using RPGM. Clearly the movable version of the protocol has better behavior because of the base stations movement.

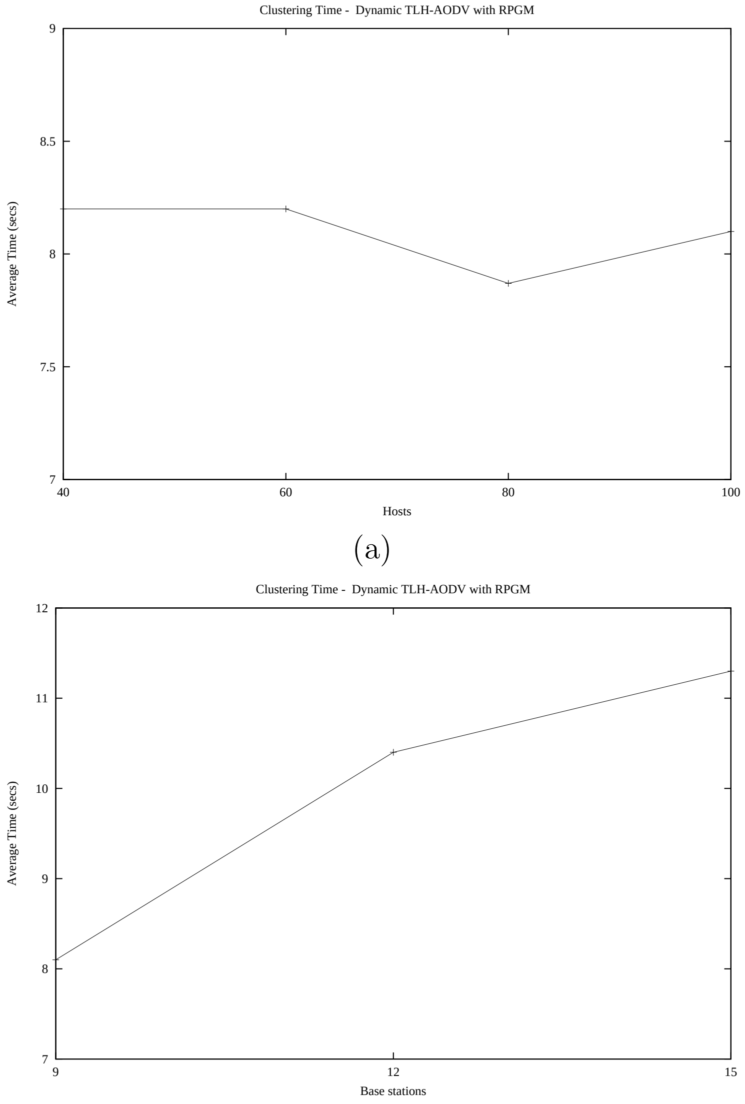
Message type	**Length (bytes)**
Hello reply	41
Hello ack	25
Enclosing box	90
Antipole	53
Termination	21
Set position	1049

Table 6.2: Length and type of messages needed to cluster and to maintain the hosts space. Hello reply: a host registers to a specific base station. Hello ack: the base station acknowledges to the hello reply. Enclosing box: message containing the enclosing box. Antipole pair: the pair sent to the base stations to split their clusters. Termination: message to stop the clustering. Set position: message sent by the root, f, to all base stations containing their new positions.

remains constant by increasing the number of hosts. Concerning the number of exchanged messages, when the number of base stations is fixed and the number of hosts varies, the number of messages remain constant except to the number of hello reply and hello ack which linearly increase. On the other hand, by varying the number of base stations and keeping fixed the number of hosts, the number of enclosing boxes, Antipole pairs and termination messages linearly increases.

Finally, simulations show that the root designation is equally shared almost among all base stations.

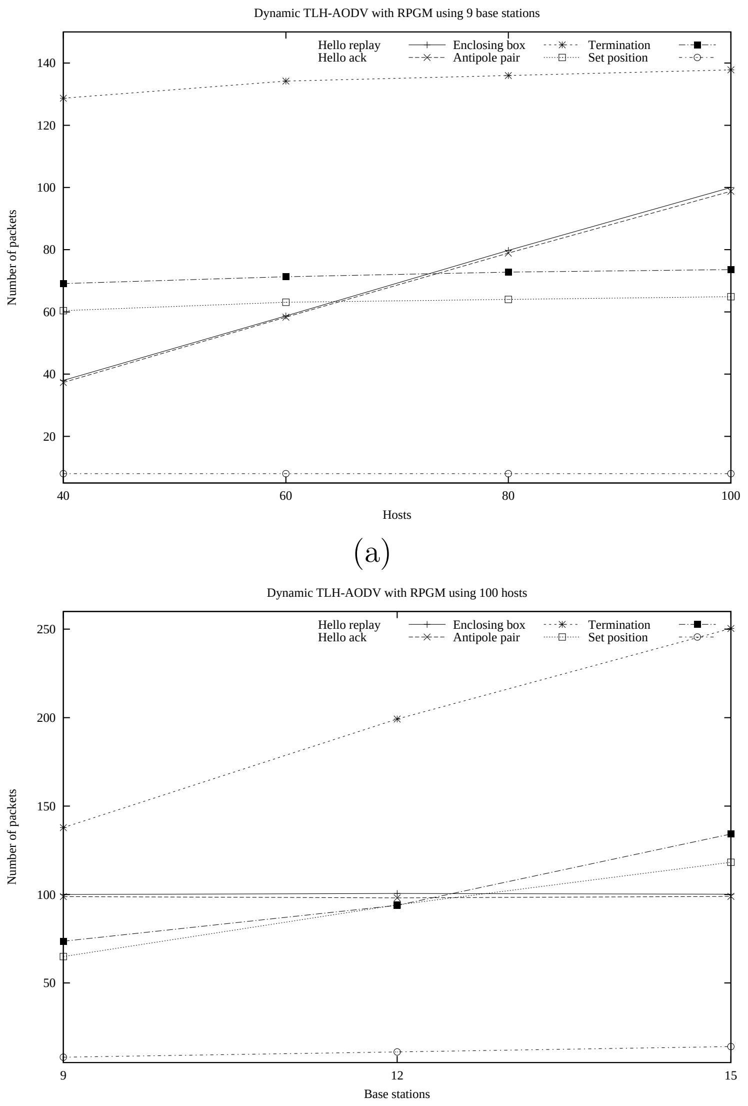
We can conclude that, the proposed protocol in connection with the realistic RPGM yields the best hosts coverage, a feasible energy consumption, and the highest percentage of delivered packets.



(a)

(b)

Figure 6.11: Time needed to cluster the hosts space.



(a)

(b)

Figure 6.12: Number of messages needed to cluster the hosts space (see Table 6.2 for messages types description).

Chapter 7

Conclusions

In this thesis, we presented a new protocol for Mobile Backbone Wireless Networks in presence of obstacles. Base stations cluster hosts through a distributed version of the Antipole Clustering algorithm and move to ensure coverage by making use of a variant of the Bottleneck Matching algorithm. Communication among hosts is guaranteed using a two level hierarchy architecture in which base station communicate through the standard AODV protocol. Hosts are assigned to base stations according to a probabilistic throughput function based on both the quality of the signal and the base station load. We implemented the protocol on top of the NS2 simulator as an extension of the standard AODV. We tested it by using the Random Way Point and Reference Point Group mobility models properly adapted to deal with obstacles. From the experiments, we can conclude that the proposed protocol in connection with the realistic RPGM yields the best hosts coverage, a feasible energy consumption, and the highest percentage of packets delivered. Furthermore, by increasing the network traffic the protocol presents the best scalability. This is essentially due to the fact that with RPGM a base station infrastructure is essential to guarantee communications.

Future research work includes distributed and approximate queries of the network such as range and k-nearest-neighbor, and messages compression. Another crucial point to investigate in the future concerns the connection of base stations. Their movement might break the network into isolated groups which could no longer communicate.

Bibliography

[1] A.Efrat, A.Itai, and M.J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31(1):1–28, 2001.

[2] P. K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM Journal on Computing*, 1999.

[3] K. M. Alzoubi, P. J. Wan, and O. Frieder. Distributed heuristics for connected dominating sets in wireless ad hoc networks. *Journal of Communications and Networks*, 4(1), 2002.

[4] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. *IEEE Infocom, Anchorage, Alaska*, (85):165–177, 2001.

[5] S. Basagni. Distributed clustering for ad hoc networks. *Proc. Int'l Symp. Parallel Architectures, Algorithms, and Networks*, 1999.

[6] P. Berkhin. A survey of clustering data mining techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

[7] C. Bettstetter. Smooth is better than sharp: a random mobility model for simulation of wireless networks. In *MSWIM '01: Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 19–27, New York, NY, USA, 2001. ACM.

[8] U. Brandes and T. Erlebach. *LNCS:Network Analysis. Methodological Foundations*. Springer, 2005.

[9] B. Brumitt, A. Stentz, and M. Hebert. Autonomous driving with concurrent goals and multiple vehicles: Mission planning and architecture. *Auton. Robots*, 11(2):103–115, 2001.

[10] 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*, 2:483–502, 2002.

[11] D. Cantone, A. Ferro, A. Pulvirenti, D. Reforgiato Recupero, and D. Shasha. Antipole tree indexing to support range search and k-nearest neighbor search in metric spaces. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):535–550, 2005.

[12] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Disc. Math.*, (85):165–177, 1990.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. MIT Press, second ed. edition, 2001.

[14] S. Cristaldi, A. Ferro, R. Giugno, G. Pigola, and A. Pulvirenti. Obstacles constrained group mobility models in event-driven wireless networks with movable base stations. *Elsevier, Ad Hoc Networks*, In Press, Corrected Proof, 2010.

[15] A. K. Datta and R. K. Sen. 1-approximation algorithm for bottleneck disjoint path matching. *Inform. Process. Lett.*, (55):41–44, 1995.

[16] A. Einstein. Investigations on the theory of brownian movement. *Dover Pubblications*, 1956.

[17] A. Ephremides, J. E. Wieselthier, and D. J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proc. IEEE*, 75(1):56–73, 1987.

[18] K. Fall and K. Varadhan. The ns manual (formerly known as ns notes and documentation). http://www.isi.edu/nsnam/ns/ns-documentation.htm. 2007.

[19] A. Ferro, R. Giugno, M. Mongiovi', G. Pigola, and A. Pulvirenti. Distributed antipole clustering for efficient data search and management in euclidean and metric spaces. *IEEE International Parallel and Distributed Processing Symposium, Rhodes Island, Greece*, 2006.

[20] A. Ferro, G. Pigola, A. Pulvirenti, and D. Shasha. Fast clustering and minimum weight matching algorithms for very large mobile backbone wireless networks. *International Journal of Foundations of Computer Science*, 14(2), 2003.

[21] E. Foudriat, K. Maly, and S. Olariu. An architecture for robust qos provisioning for mobile tactical communications. *MILCOM 2000 - IEEE Military Communications Conference*, 1:85–89, 2000.

[22] M. Gerla, T. J. Kwon, and G. Pei. On demand routing in large ad hoc wireless networks with passive clustering. *IEEE Wireless Communications and Networking Conference*, (1):100–105, 2000.

[23] M. Gerla and J. T. C. Tsai. Multicluster, mobile, multimedia radio network. *ACM-Baltzer Journal of Wireless Networks*, 1(3):255–265, 1995.

[24] M. Gerla, K. S. Xu, and A. Moshfegh. Minuteman: Forward projection of unmanned agents using the airborne internet. *IEEE Aerospace Conference*, 2002.

[25] M. Gerla and K. Shin Xu. Minuteman: Forward projection of unmanned agents using the airborne internet. *IEEE Aerospace Conference*, 2002.

[26] X. Hong, M. Gerla, G. Pei, and C. Hiang. A group mobility model for ad hoc wireless networks. In *MSWiM 99: Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 53–60, New York, NY, USA, 1999. ACM.

[27] Teerawat Issariyakul and Ekram Hossain. *Introduction to Network Simulator NS2*. Springer, 2009.

[28] A. Jardosh, E.M. Belding-Royer, K.C. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 217–229, New York, NY, USA, 2003. ACM.

[29] A.P. Jardosh, E.M. Belding-Royer, K.C. Almeroth, and S. Suri. Real-world environment models for mobile network evaluation. *Selected Areas in Communications, IEEE Journal on*, 23(3):622–632, March 2005.

[30] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing, T. Imelinsky and H. Korth, editors,Kluwer Academic Publishers*, pages 153–181, 1996.

[31] D.B. Johnson, D.A. Maltz, and J. Broch. Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*, pages 139–172, 2001.

[32] M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. In *Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Barcelona, Spain, April 2006.

[33] T. J. Kwon and M. Gerla. Clustering with power control. *IEEE Military Communications Conference*, (1):1424–1428, 1999.

[34] C. R. Lin and M. Gerla. Adaptive clustering for mobile wireless networks. *IEEE Journal on Selected Areas in Communications*, (16):1265–1275, 1997.

[35] Y. Lu, W. Wang, and B. Bhargava. Hierarchical structure for supporting movable base stations in wireless networks. *IEEE International Conference on Telecommunications*, 2003.

[36] A. B. McDonald and T. Znati. A mobility based framework for adaptive clustering in wireless ad-hoc networks. *IEEE Journal on Selected Areas in Communications*, 17(8), 1999.

[37] M. Musolesi, S. Hailes, and C. Mascolo. An ad hoc mobility model founded on social network theory. In *MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 20–24, New York, NY, USA, 2004. ACM.

[38] NS2 The network simulator. http://www.isi.edu/nsnam/ns/. 2009.

[39] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. *ACM/SIGCOMM 1994. Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[40] C. E. Perkins and E. M. Royer. The ad hoc on-demand distance vector protocol. *Ad hoc Networking*, pages 173–219, 2000.

[41] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, 1985.

[42] T. S. Rappaport. *Wireless Communication: Principles and Practices.* Prentice Hall, 1996.

[43] M. Royer and C-K Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *In IEEE Personal Communication Magazine*, 6:46–55, 1999.

[44] A. Srinivas and E. Modiano. Joint node placement and assignment for throughput optimization in mobile backbone networks. *INFOCOM 2008. 27th IEEE International Conference on Computer Communications*, pages 1130–1138, 2008.

[45] A. Srinivas, G. Zussman, and E. Modiano. Mobile backbone networks: construction and maintenance. *MobiHoc 2006. Proceedings of the 7th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing*, pages 166–177, 2006.

[46] R. Snchez, J. Evans, and G. Minden. Networking on the battlefield: Challenges in highly dynamic multi-hop wireless networks. *In Proc. of IEEE MILCOM*, 1999.

[47] J. Urrutia. Art gallery and illumination problems. *J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry*, pages 973–1027, 200.

[48] X. Wang and K. Kar. Throughput modelling and fairness issues in csma/ca based ad-hoc networks. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 23–34, 2005.

[49] K. S. Xu and M. Gerla. A heterogeneous routing protocol based on a new stable clustering scheme. volume 2, pages 838–843 vol.2, Oct. 2002.

[50] X. Zhang and G. F. Riley. Performance of routing protocols in very large-scale mobile wireless ad hoc networks. *Modeling, Analysis, and Simulation of Computer Systems, International Symposium on*, 0:115–124, 2005.