Tracking the node path in wireless ad-hoc network

Artur Sierszeń
Lodz University of Technology,
Institute od Applied Computer
Science, ul. Stefanowskiego 18/22,
90-924 Łódź, Poland
Email: artur.sierszen@p.lodz.pl

Łukasz Sturgulewski
Lodz University of Technology,
Institute od Applied Computer
Science, ul. Stefanowskiego 18/22,
90-924 Łódź, Poland
Email:
lukasz.sturgulewski@p.lodz.pl

Agnieszka Kotowicz
Lodz University of Technology,
International Faculty of
Engineering, ul. Żwirki 36, 90-924
Łódź, Poland
Email:
agnieszkakotowicz88@gmail.com

***Abstract*—This article provides an insight into the topic of ad-hoc protocols used for routing, namely proactive and reactive protocols. It depicts the general concept how these protocols can find a path in a network between two nodes and it also presents the evaluation of the methods of tracking the node path in a wireless ad-hoc network through investigating the available mobile routing protocols.**

The main focus is on the throughput and the average end-to-end delay in a network, using for the simulation OMNeT++ environment. Three protocols were chosen for the final testing: Ad-hoc On-demand Distance Vector (AODV), Optimized Link State Routing (OLSR), and Dynamic Source Routing (DSR).

I. Introduction

AD-HOC networks originated in 1960s when the ALOHA project was emerging from the shadows. Even though the dynamically established network was not the first outcome of this project (it was based on fixed nodes with the single-hop option only), the idea of a shared medium for client transmissions remained. The earliest wireless ad-hoc networks were the “packet radio” networks (PRNETs) already proposed in 1970. Since then, project and ad-hoc networks have been developed continously.

In general, an ad-hoc network is a collection of wireless mobile nodes (e.g. smart phones, laptops, cameras etc.) that is formed only for a short period of time when wireless devices come within each other’s communication ranges. Nodes are the users or devices forming the network [1]. This set-up is created dynamically without using a preconfigured network infrastructure (a simple example of an ad-hoc network is shown in Fig. 1). If a network is set up for a longer period of time, it is just a plain old local area network (LAN). Finally, it is said that an ad-hoc network does not have any centralized architecture, what means that any node is a peer. In a peer network, each node is a client, a receiver (server), or a mediator of a packet that routes packets to other nodes that are out of range of the sender [2]. Moreover, an ad-hoc network can operate as a stand-alone, closed group as well as a network with a connection to the Internet.

This definition indicates that the mobility of the nodes leads to fast and sometimes enormous changes in the wireless network topology. In addition, other obvious attributes such as a large size of the network, bandwidth, large diversity of available devices, and their power consumption may cause large problems for today’s routing protocols. They may all be a huge challenge if ad-hoc network users want to receive a reliable and high quality service, not to mention other problems that can be enumerated: physical obstacles, indirect communication between two nodes, imperfections of network elements causing delays, battery constraints etc.

The first idea is based on fast routing protocols. User mobility influences the changing topology of an ad-hoc network, so it is possible that some old nodes are no longer available but new have just appeared. In theory, a routing protocol could still handle this change somehow in order to connect the required nodes, but there are some protocols that cannot do that. Therefore, it is necessary to use the protocols that are dedicated for ad-hoc networks and, providing they are fast enough, they can solve the mobility problem. Routing in ad-hoc networks is a combination of dealing with topology adjustments and minimizing the routing overhead. There are proactive and reactive protocols as well as the hybrid of those two solutions which tries to combine the best features of each protocol [1].

In networking, a hop represents one fragment of a path between the source and the destination. It is a well-known phenomenon that data passes through an unknown number of intermediate gateways until it reaches its destination. For example, on the Internet packages are routed between various sub-networks. Moreover, the definition of a hop distance should be useful. It is a unit of measurement used to express the number or routers that a packet must pass through on its way to its destination.

Therefore, in a wireless network, single-hop means that there is only one hop between the source station and the destined host. At the same time, multi-hop refers to a situation when the packet of data must travel through more than one hops. The hop count is important for the basic network operating principles. Fig. 1 clearly presents both expressions.

The perfect routing protocol has to combine the goal of dynamic adjustment to changing conditions in an ad-hoc network and of low overhead. Due to this combination, several different approaches were introduced in the field of routing protocols. Some of them will be presented and discussed in the following sections. Figure 2 shows a possible classification of routing protocols that is taken from Latiff et al. [3].

978-1-4673-4471-5/$25.00 © 2013, IEEE

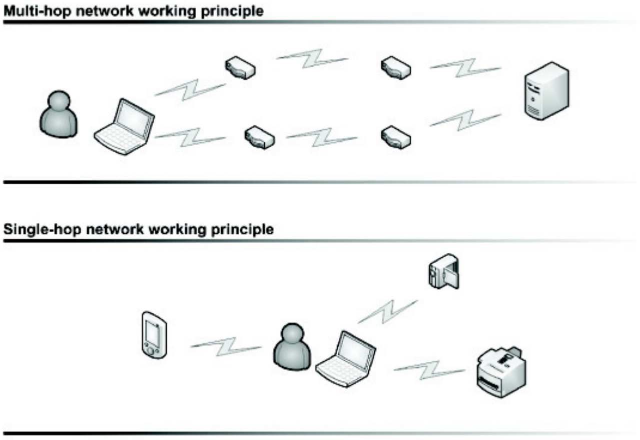
Multi-hop network working principle
Single-hop network working principle

Fig. 1 Examples of single-hop and multi-hop ad-hoc networks.

II. Flooding

According to Mohapatra and Krishnamurthy [4], flooding (network-wide broadcasting or pure flooding) is a way to deliver data from the source node to the destination node through every outgoing link. It means that every attached node will receive source data packets via a MAC layer broadcast mechanism and, finally, every node in the connected component of the network will deliver the data.

There is the basic rule that is followed in order to avoid looping in the network: "every node transmits only once". If a node collects data for the first time, it re-broadcasts it. This algorithm guarantees the end of the procedure eventually and is easy to implement. Additionally, no prior knowledge about the network topology is required and, in some cases, when mobility of the nodes in the network is so high that even unicast protocols cannot handle it, the flooding may become the only reasonable alternative for routing data rationally [4].

This protocol technique can have a big contribution to an overall throughput in the network – the higher number of packets in a network means that there is a higher chance for a collision, what influences the success rate of the packet delivery directly.

III. Proactive Protocols

The main operating principle of proactive routing protocols is that they maintain unicast paths between all pairs of nodes, even when routes are currently idle. They are also called "table-driven" routing protocols. A node can decide to update its routing table after either receiving an update message from a neighbor or detecting a change in the status of a link to a neighbor. Hence, when the source wants to start a connection with a remote destination node, the process can immediately begin because the path is ready and available at any time. No other request or path discovery is required and, therefore, the delay of such nature can be eliminated. It is assumed that the protocols are capable of finding the shortest and the most optimal route for a given model of link costs.

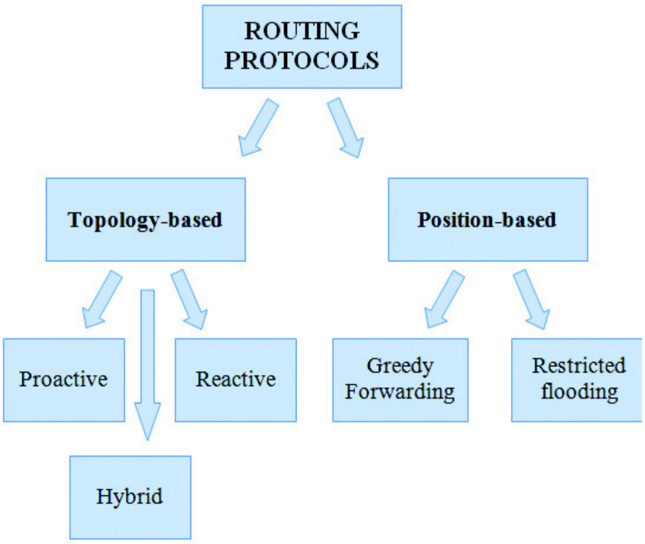

Fig. 2 Categorization of MANET routing protocols.

Optimized Link State Routing (OLSR) is a member of the proactive protocols group and, as such, it is also a table-driven protocol, which assumes that nodes in an ad-hoc network will update each other regularly and will cooperate in order to send data from the source to the destination using the most optimal path. This protocol uses the Mohapatra and Krishnamurthy [4] concept of Multipoint Relays (MPRs), mentioned in this paper before.

In a general operating mechanism, only the nodes that were selected as responsible for their area are allowed to generate link state updates. Additionally, these updates must include information on the links between MPR nodes only [2]. No other node has been granted the privilege to do so in order to keep the update size as small as possible. This way, even though there may be other routes available, only a part of the network topology is revealed to other nodes. This may seem dangerous for network routing; however, this partial information is fully sufficient in order to locally calculate the hop count to every node because it is certain that a path that consists only of MPRs exists.

One of basic principles of OLSR is that it uses only periodic updates in order to keep all nodes up-to-date with the link state. Whenever traffic in a network is dense, the protocol reduces the overhead as compared to the time when traffic is lower or the network is sparse. Additionally, the interval between subsequent updates is critical for reacting to topology changes and should be accurately considered.

What is really distinguished for OLSR is that it can minimize the overhead from flooding of control traffic effectively only by using carefully selected MPRs to retransmit control messages. This way, not all nodes have to be occupied with retransmission but messages still reach all nodes in an ad-hoc network [4]. Moreover, in order to find the most optimal route, the OLSR protocol needs only a partial link state to be sent through the whole network. This minimal information about link states includes the links to all nodes in the region under MPR responsibility (however, the redundancy is also possible).

The performance of the OLSR protocol was also tested in comparison to other various types of protocols. It was discovered that OLSR shows a good resilience to a suboptimal link state situation in a network where the routes are constantly changing (as nodes move in counter-rotating circles) and the network picture never converges permanently [5].

IV. Reactive Protocols

Reactive routing protocols, also called "on-demand" protocols, are quite different from the traditional proactive manner. The main difference lies in a route preservation mechanism – while proactive protocols keep all routes available for use at any time, reactive protocols maintain only the paths that are currently needed. The advantage of this technique is that a huge amount of routing data does not have to be stored and updated all the time. However, good algorithms are needed for instant path discovery that would not create too big delays and queues. Still, this kind of protocols should be perfect for networks where the traffic is small and sporadic.

Ad-hoc On-demand Distance Vector [6] belongs to the reactive protocols family and discovers a route from the source to the destination when it is needed. All possible routes are not maintained the whole time. Basically, AODV relies on the distance vector technique. This term refers to the method which uses the arrays of distances to other nodes in a network. Instead of saving knowledge about all routes in a network, it is enough to know the direction of forwarding the message (or the interface that should be used) or the distance from its destination (in reasonable units).

So, keeping those two basic rules in mind, AODV depends on dynamically established route table entries at nodes between the source and the destination. This means that AODV protocol requires a much larger overhead in order to piggyback source routes in each packet, what is unthinkable for proactive protocols. Each entry consists of the destination address, the next hop address, the destination sequence number, and the hop count.

Another characteristic of AODV protocol is the sequence number, which is incremented monotonically at each node of the network separately. The combination of above features results in an algorithm that can use an available bandwidth efficiently and can adapt to changes spotted in an ad-hoc network.

Each router based on the AODV protocol is more or less a state machine that works using a simple algorithm. If a route exists, then the message is forwarded. Otherwise, the message enters a queue and the router sends a route request in order to search for a possible path. According to received information, the router can update the table and even transmit the message if the path to the destination has built up.

The AODV protocol uses four types of messages that the nodes can distinguish [4]. The route discovery is handled by Route Request (RREQ) and Route Reply (RREP). The needed paths are maintained by Route Error (RERR) and HELLO messages.

Dynamic Source Routing is the reactive protocol that uses a source routing mechanism. The sender of the packet generates a header that can contain all addresses of the nodes in a network which a packet must be forwarded through in order to reach the destination node [7]. It means that the source needs to know the whole hop-by-hop path that can be stored in a route cache. This memory should be maintained by each node of an ad-hoc network which wants to participate in the traffic share. If this cache does not enclose the required path, the node simply needs to use the standard discovery process for the wanted route in order to dynamically determine the path to the destination node. It is accomplished by flooding the network with RREQ messages, also called queries [4].

The route discovery technique is based on route requests which are re-broadcasted by each intermediate node if it is not a destination node or if it does not know the path to the destination based on a route cache. Otherwise, the node answers with the PREP message and the packet with the entire route is sent back to the origination node. And finally, this path is, of course, saved for later in a route cache by each node which does not know it [4]. Like in the case of the AODV protocol, the RERR packet is generated if any node detects a broken link that cannot be longer used for the traffic. This kind of message triggers the removal of the given route from the route cache as well as all entries that are affected [4].

V. Running Simulations

We have decided to use OMNeT++ [8] for the simulations library, which is rather a good provider of infrastructure and tools than a simple simulator of a network. The main advantages of this tool are as follows: free for academic use, the engine runs event-driven simulations of communicating nodes on a wide variety of platforms, support of graphical network creation, the framework is fully extensible and modular (based on C++ language), the documentation and the tutorials are properly maintained and developed by an increasing number of new users, and there exists a great diversity of available libraries and featured projects compatible with the OMNeT++ platform.

We performed the tests only in a random grid, but we were aware of the disadvantages of the linear or grid topology that may cause problems in ad-hoc network routing. Nevertheless, we wanted to have a more realistic topology, so the random one was the most reasonable (Fig. 3). All the nodes in our simulations were moving all the time with variable speed and direction of movement without a pre-determined path.

All tests had the common goal of adjusting the final simulation parameters because the default ones are not always the most suitable for the wanted results. Table (Table I) presents the output of all testing and verification processes.

The preparation for simulation was not only focused on investigating the best parameter set but also on adjusting the behavior of a singular node. The inner construction of the mobile device was based on the TCP/IP model. It was decided that it would employ 802.11g technology for MAC layer, UDP was used in transport layer, and UDP APP for application layer.

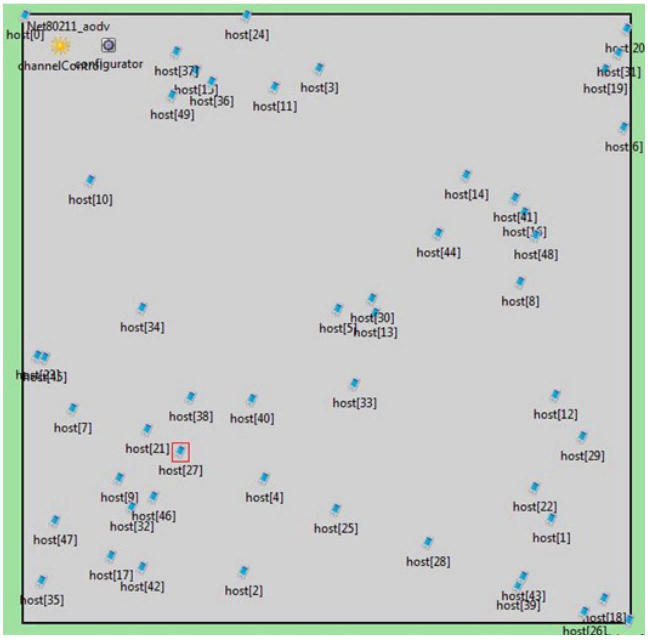

Fig. 3 Snapshot of the random topology.

VI. Results

With all predefined parameters of the simulations, we were able to obtain the relevant output – the average end-to-end delay and the average throughput of the network. The average delay time involves all possible reasons, such as queuing time, packet transmission, and propagation time or retransmission time.

We think that the delay is important for a dynamic ad-hoc network and should be as small as possible but, at the same time, the successful rate must be tolerable. Please have a look at the results that we got separately for the AODV, OLSR and DSR protocols (the plots are presented in Fig. 4, 5 and 6, respectively).

With the increased number of nodes in a network, packet collisions may occur more often and this will lead to a higher number of retransmissions. This kind of situation would definitely influence the overall delay and it can be observed in all of these plots.

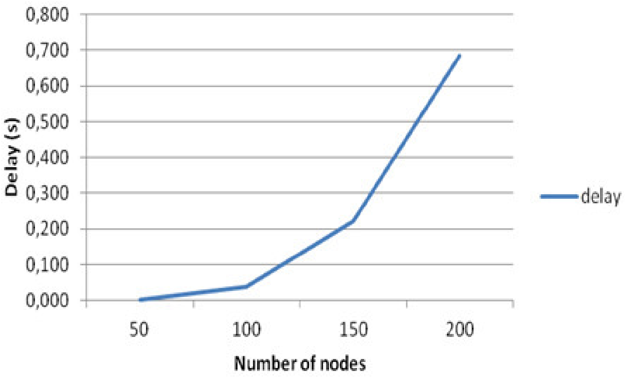

Fig. 4 Average end-to-end delay for the OLSR protocol.

Table I.
Simulation parameters chosen for the final tests.

Parameter	Value
Simulation time	600 s
Topology	Random location of nodes (network of mobile devices)
Number of nodes	50, 100, 150, 200
Ad-hoc protocols	OLSR, AODV, DSR
Transmission range	100 m
Mobility model	Random way-point

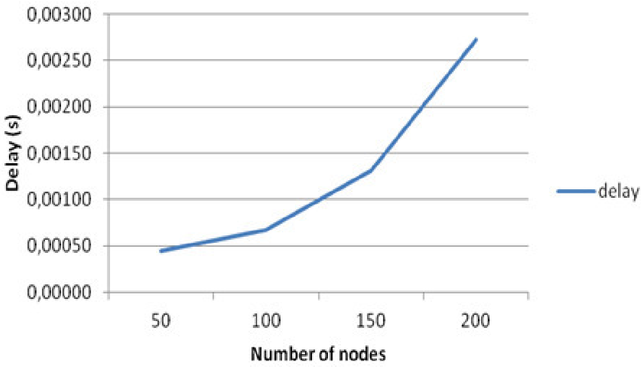

Fig. 5 Average end-to-end delay for the AODV protocol.

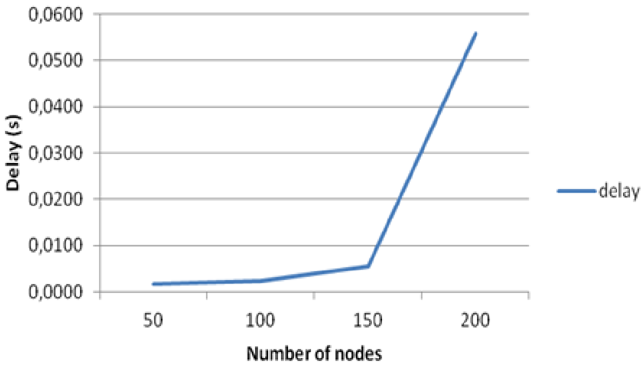

Fig. 6 Average end-to-end delay for the DSR protocol.

However, the smallest delay can be noticed in case of the AODV protocol application.

The average throughput of all three protocols is compared by measuring the average rate of successful message delivery over a communication channel. This is calculated in bits per second, what emphasises the vitality for ad-hoc network operation. The higher this number is, the higher the throughput is.

When the number of nodes increases, more packets of data come to the network; it can be observed that the highest throughput of all three investigated protocols was reported in the AODV protocol (please compare the plots presented in Fig. 7, 8 and 9).

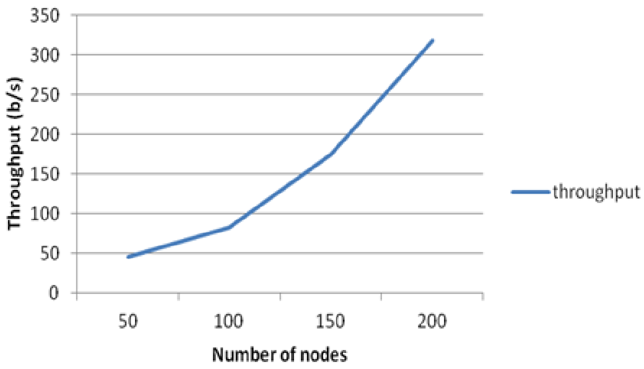

Fig. 7 Average throughput for the OLSR protocol.

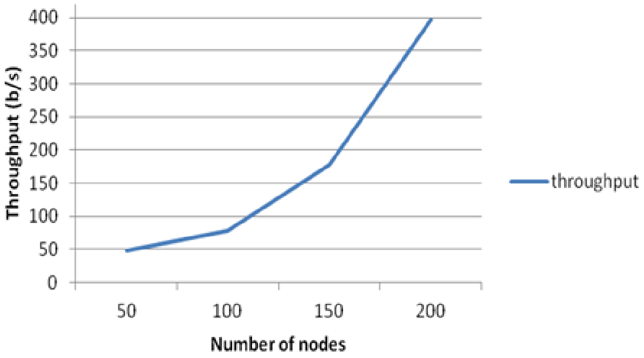

Fig. 8 Average throughput for the AODV protocol.

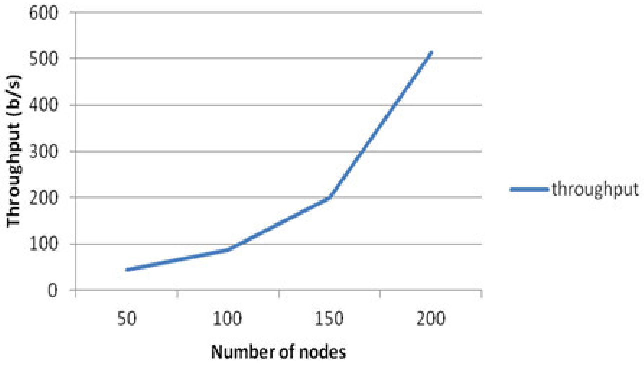

Fig. 9 Average throughput for DSR protocol.

VII. Discussion

We have examined and analyzed three routing protocols from both proactive and reactive groups, namely Ad-hoc On-demand Distance Vector (AODV), Optimized Link State Routing (OLSR), and Dynamic Source Routing (DSR). The simulation tests involved measuring metrics such as throughput and end-to-end delay.

The presented results indicate that the performance of the AODV protocol is superior as compared to the two other protocols that have been taken into account. It can be easily noticed that the AODV protocol handles the network traffic better when more and more nodes are added to an ad-hoc network. Still, according to the theoretical background, smaller networks (up to 10 nodes) might have been handled better by the DSR protocol.

Nevertheless, the DSR protocol was inferior in both end-to-end delay and throughput, even when there were only 50 nodes in the network. The poor performance of DSR with respect to time for packet delivery is mainly due to caching the routes and the lack of mechanism for deleting the stale paths. We think that it is the reason why DSR did not perform so well, even though it belongs to the same group of reactive protocols as the AODV protocol.

It was also observed that, for a relatively small number of nodes, all routing protocols are similar when throughput was under test. However, the average end-to-end delay could be easily compared for even the smallest number of nodes and the AODV protocol was incredibly fast in delivering packets. This is why we assume that the AODV protocol would be preferred for real time traffic over DSR or OLSR.

Whenever throughput is considered, results show that DSR does not handle it well because it consumes a considerable amount of power. If it was a real environment involving mobile devices, the batteries would run out of power pretty quickly.

During the testing process, some problems occurred and those influenced directly the developing and testing time. The parameters were difficult to adjust because the default ones did not give the wanted results and looking for better values consumed a lot of time. Additionally, testing the performance of the DSR protocol was difficult in terms of the processing power of computer (this protocol uses the caching routes technique, so the bigger the number of nodes was, the more resources it was consuming for the test). Finally, the OMNeT++ testing environment gives different debugging messages in each operating system, so we had difficulties solving, for example, Cygwin problems during the testing process.

References

[1] R. Hekmat, Ad-hoc Networks: Fundamental Properties and Network Topologies, Springer, 2006

[2] S. K. Sakar, T.G. Basavaraju, C. Puttamadappa, Ad-hoc Mobile Wireless Networks. Principles Protocols and Applications, Auerbach Publications, 2008

[3] L.A.Latiff, N. Fisal, S.A. Arifin and A. Ali Ahmed, Directional Routing Protocol in Wireless Mobile Ad Hoc Network, article from "Trends in Telecommunications Technologies", book edited by Christos J Bouras, 2010

[4] P. Mohapatra, S. V. Krishnamurthy, AD-HOC NETWORKS: Technologies and Protocols, Springer Science, 2005

[5] J. Hsu, S. Bhatia, M. Takai, R. Bagrodia, M. J. Acriche, Performance of mobile ad hoc networking routing protocols in realistic scenarios, IEEE Military Communications Ceonferenc, MILCOM 2003, 2003

[6] C. E. Perkins, E. M. Royer, Ad Hoc On-Demand Distance Vector Routing, In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), 1999

[7] D. Johnson, D. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile computing, 1996

[8] OMNeT++ Network Simulation Framework, www.omnetpp.org