A Framework for the Management of Large-Scale Wireless Network Testbeds

Krishna N. Ramachandran, Kevin C. Almeroth, Elizabeth M. Belding-Royer
Department of Computer Science
University of California
Santa Barbara, CA 93106
{krishna, almeroth, ebelding}@cs.ucsb.edu

***Abstract*— Wireless testbeds are typically distributed over large physical areas. There are often many nodes, some of which are difficult to reach on-site or via remote access. As a result, such nodes may be manageable using only in-band management techniques making the task of testbed management challenging. As a remedy, we propose the *ATMA* framework, a framework which enables out-of-band management by deploying a multi-hop mesh network alongside a testbed to manage the latter. The ATMA mesh network is designed to be self-configuring and therefore can be installed with minimal effort. As an extension of the ATMA framework to multi-hop wireless testbeds, we have designed and developed a suite of tools for the management and monitoring of multi-hop wireless testbeds. This paper presents the design of the ATMA framework, its extensions, and describes our implementation of the framework using low-cost, commodity wireless devices.**

I. Introduction

The primary platforms for conducting wireless network research in recent years have been simulation tools such as NS-2 [12] and Glomosim [18]. Although simulation tools facilitate a speedy and controlled evaluation of a solution, they fail to accurately represent "real-world" network characteristics such as fluctuating link qualities and wireless interference [13]. As a result, solutions evaluated via simulations may not conform to predictions upon deployment. Therefore, there is a growing consensus among researchers to evaluate solutions using network testbeds.

A thorough testbed-based evaluation is strong evidence that a solution performs as expected in real-world deployments. Performing a testbed-based evaluation, however, can be challenging because testbed installation, configuration, and management are labor-intensive operations. One reason for the inefficiency is that, because wireless testbeds are distributed in areas that may be large, remote, or inhospitable, "out-of-band" access to the nodes, i.e., where the nodes in the testbeds are accessible via a separate wired/wireless management interface, may not be feasible. Consequently, tools to manage such testbeds must rely on "in-band" techniques where the management traffic competes with the testbed traffic. In-band management has the drawback that faults in the operation of the testbed itself can result in its nodes being inaccessible and therefore not manageable. In-band management also adds overhead to the traffic flowing in the testbed.

In this paper, we present the *ATMA* framework, a testbed management framework designed to address the lack of techniques to gain out-of-band access to testbed nodes deployed in large, remote, or inhospitable terrains. Using this framework, a multi-hop mesh network can be installed alongside a testbed to enable out-of-band management. The mesh network is designed to be self-organizing and therefore can be installed by a testbed operator with minimal configuration. Because of the continuing decline in price of wireless hardware, deploying such a mesh network should be cost-efficient.

As a proof-of-concept, we have implemented the ATMA framework using low-cost Linksys WRT54G wireless devices and a multi-path version of the Ad hoc On-Demand Distance Vector (AODV) routing protocol [14]. We have also developed, as an extension of the framework, a suite of tools to manage and monitor multi-hop wireless testbeds. The framework implementation and its extensions are being used to manage a twenty-five node multi-hop wireless testbed deployed on the UC Santa Barbara campus.

The remainder of this paper is organized as follows. Section II presents our motivation for the ATMA framework in more detail. In Section III, we present the architecture of the framework. In Section IV, we describe the extensions to the ATMA framework to help manage and monitor multi-hop wireless testbeds. Section V briefly discusses our implementation of the framework, and Section VI concludes this paper.

II. Motivation

Wireless networks are deployed in large areas such as housing communities and multi-storied buildings, as well as in remote and inhospitable regions such as forests, swamps, arctic areas and deserts. Each of these target deployment environments has diverse network characteristics. Key characteristics include the reachability of wireless links, traffic patterns, and interference. As an example of such diversity, line-of-sight between wireless devices, typically achievable in "roof-top" wireless networks [3], can provide long-range reachability compared to non-line-of-sight wireless links deployed inside buildings. To design protocols and systems for these different deployment environments, it is necessary to evaluate them on testbeds that closely resemble the target deployments.

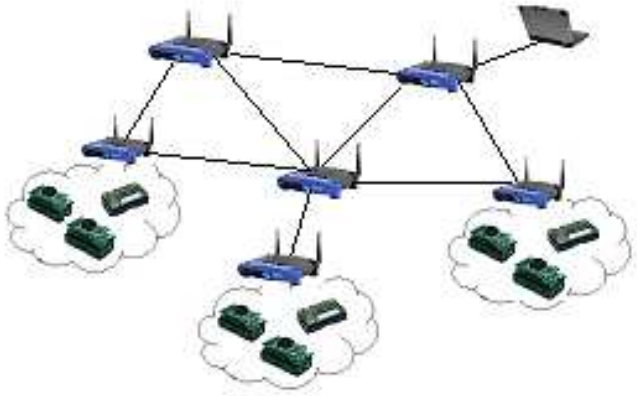

Fig. 1. Architecture of a support mesh for a sensor network testbed.

Managing network deployments "in-band", i.e., where the management traffic is a portion of the overall testbed traffic, however has two main drawbacks. One, in-band management is challenging because tools to configure the network must issue commands in a precise and well coordinated manner. To motivate why such behavior is required, consider an example where all nodes in a multi-hop wireless network are required to switch to a new operating frequency. Tools that perform the frequency switch must ensure that the switching starts from the periphery of the network and not from within. Random changes of the node frequency assignment will likely result in the nodes at the periphery becoming unreachable and the network becoming disconnected. Even knowing which nodes are at the periphery of the network, so that frequency changes can be planned, is a difficult task. The design and implementation of tools that can handle such cases is challenging. Moreover, such tools cannot overcome network disconnections that occur because of the (faulty) operation of the testbed itself. This can happen, for example, if there is an implementation related bug or a flaw in the design of a protocol used in the testbed.

The second drawback is that in-band management adds to the traffic overhead in the testbed. This overhead can be significant in cases where frequent reconfiguration of the network takes place or continuous monitoring of the testbed is performed. For example, wireless network monitoring tools [15], [11], [20] send collected information, such as packet logs and traffic statistics, to a monitoring sink on a frequent basis, thereby adding to the traffic overhead in the testbed. This overhead can result in an inaccurate analysis of the performance of the testbed.

As a result of the disadvantages of in-band management, "out-of-band" management of testbeds, where the nodes in the testbeds are accessible via a separate wired/wireless management interface, is an attractive alternative. Out-of-band management, however, may not be feasible when remote access to the testbed nodes is limited due to environmental conditions or logistical reasons. As an example, we recently deployed a twenty-five node multi-hop network in a building on the campus of UC Santa Barbara. Of the five floors of the building on which nodes are present, only nodes on three floors of the building are reachable via one of two "access" networks, one wired and the other "WiFi"; both the access networks have coverage only on the first three floors of the building. The remaining nodes on the last two floors are manageable only "on-site", which is a labor intensive operation. As another example, consider a testbed deployed in an inhospitable region such as a swamp or a forest where out-of-band access to the testbed is typically not available.

To overcome the challenge of gaining out-of-band access to testbed nodes, we propose the use of a multi-hop wireless mesh network that is deployed alongside the testbed to assist in out-of-band management. In such a support mesh network, a testbed manager issues management commands to its agents, and the commands reach the agents over multiple hops in the support mesh. This alleviates the lack of out-of-band access to testbed nodes. Figure 1 illustrates our proposed solution. In the figure, a mesh network is deployed alongside a sensor network testbed. Each mesh node manages a cluster of sensors. The testbed operator uses the management station to send instructions to the agents. The instructions are carried over multiple hops to the mesh nodes.

Because of the falling price of portable wireless devices, deploying a support mesh network is cost-efficient. For example, a cheap commodity wireless device that is also capable of running mesh software is the Linksys WRT54G device. The use of such a device to setup a support mesh network enables a viable, cost-efficient solution for testbed management.

For the support mesh network to be of maximum benefit, it should require minimal configuration during deployment. Therefore, a critical requirement of the mesh deployment is that it *self-configure*. By self-configuration, we mean that the nodes in the mesh automatically discover network specific parameters such as the wireless channel, the network name (ESSID), and the IP address in order to communicate in the mesh.

As a final motivation, there has been significant recent work on the use of testbeds to facilitate research. Experiments conducted on testbeds created by the Orbit Project [16], MIT Roofnet [5], [3], and Microsoft Research [6], [7] require continuous access to the testbed nodes. Other testbed-based

experiments conducted by Dartmouth [8], UCSB [1], Uppsala University [10], and USC [19] have required tools to manage their testbeds "in-band". These projects likely would have benefited from the ATMA management framework. Beyond these testbeds, we are not aware of any work similar to the ATMA framework. Given the need for the ATMA framework, we now focus on presenting it in more detail.

III. ATMA Framework Architecture

This section describes the ATMA framework. An overview of the framework is presented first followed by a more detailed description.

A. Overview

The ATMA framework is based on a typical client-server architecture. An agent (client), co-located with a testbed node, collaborates with other agents in the network to form an ATMA mesh. The agents use the newly formed mesh to communicate with a centralized manager (server). The manager has a complete view of all agents in the mesh. It provides a management interface via which an operator can issue commands to the agents, and the agents in turn control the testbed nodes. To minimize control traffic overhead, a reactive routing protocol is the preferred choice for routing within the mesh, although a proactive protocol could also be used.

To minimize the overhead in deploying the ATMA mesh, agents should self-configure in order to automatically create the mesh. To achieve this, the following three steps are performed within the ATMA mesh:

- *Manager Beaconing*: To facilitate the automatic discovery of the manager, the manager sends periodic beacons to advertise its presence.
- *Agent Boot-Strapping*: An agent boot-straps itself by discovering network specific parameters such as the network name, wireless channel for communication, and the IP address to use in the mesh.
- *Agent Registration*: Once an agent boot-straps itself, it registers with the manager. At this point, the manager can issue commands to the agent to additionally configure it as may be required.

The remainder of this section discusses the above steps in more detail. In our discussion, we assume without any loss of generality that the ATMA mesh is created using commodity IEEE 802.11 wireless devices.

B. Manager Beaconing

Periodic beacons sent by the manager facilitate the automatic discovery of the manager by the agents. These beacons are re-broadcast throughout the ATMA mesh by the agents in the network. As we explain later in this section, the beaconing also serves to synchronize all agents to the same BSSID cell[1].

[1]The BSSID is a field in the IEEE 802.11 management header. Nodes are required to be part of the same BSSID cell in order to communicate with each other.

The hop-by-hop propagation of beacons can be performed using either a MANET broadcast protocol [9], [17] or by simply overloading the semantics of a routing protocol control message that is transmitted frequently by an agent. As an example of such a message, MANET routing protocols like AODV [14] use a periodically broadcast control message to detect neighbors. By embedding the beacon information in such messages, the overhead of flooding beacons in the network can be minimized.

C. Agent Boot-Strapping

An agent needs to boot-strap itself to communicate on the ATMA mesh. As a first step, the agent binds its network interface to an IP address. The second step involves discovering network related parameters such as the wireless channel number and the network name of the ATMA mesh.

To bind its network interface to an IP address, the agent picks a temporary IP address randomly from the auto-configuration IP address range (169.254/16) [4]. This IP address range has been set aside by the Internet Engineering Task Force (IETF) standards body for the creation of instantaneous ad hoc networks that lack an address resolution authority.

Upon binding an IP address to its network interface, the agent scans all channels for active wireless networks[2]. Upon finishing the scan, the agent joins each discovered network temporarily to check if the network is the ATMA mesh. This check is possible because of the periodic beacons broadcast throughout the ATMA mesh. If an agent receives a beacon within a time interval equal to three times the beacon re-broadcast interval (to tolerate beacon losses), it concludes that it has joined the ATMA network. If the agent does not receive a beacon within the sampling interval, it concludes that the currently joined network is not the ATMA mesh and joins the next network in the scanned list of networks.

The beacons also serve to synchronize all the agents to the same BSSID cell as the manager. This is required because the unpredictable nature of wireless packet reception can result in the mesh partitioning into different BSSID cells and consequently becoming disconnected. As a solution to this problem, an agent initiates the above described channel scan procedure (if it does not receive a beacon on the network it has joined) with the goal of discovering the correct BSSID cell. In our implementation, the time interval after which the agent initiates the scan is equal to three times the beacon re-broadcast interval to allow for any beacon losses.

D. Agent Registration

Immediately upon discovering the correct network parameters for the ATMA mesh, an agent needs to first ensure that the IP address it has chosen is unique in the mesh. Otherwise duplicate IP addresses within the mesh can result in incorrect packet delivery to the agents. To perform the unique IP address check, an agent sends a REGISTER-REQUEST message to the manager. The manager, upon receiving the

[2]It is assumed that the network name for the ATMA mesh is contained in the IEEE 802.11 management header.

request message, waits for a period equal to the route timeout period before sending a REGISTER-REPLY message. The wait period is required to remove any cached entries for the originating agent in the routing tables of intermediate nodes. Once the wait period is over, the manager broadcasts a ROUTE-DISCOVERY message for the agent's IP address. If the manager receives ROUTE-REPLY messages from multiple agents[3], it does not send the REGISTER-REPLY message. If an agent does not receive its REGISTER-REPLY message within a timeout period, it concludes either that it has chosen a duplicate IP address or the REGISTER-REPLY has been dropped at an intermediate node for some reason. In both cases, the agent selects a different IP address before re-sending the register message.

Upon successfully receiving a REGISTER-REPLY, the agent has finished registering with the manager. The agent can then request the manager to send it other configuration information, such as its permanent IP address, so that it can reconfigure as per the requirements of the testbed operator.

IV. ATMA Management Tools

In this section, we describe four tools specifically developed as an extension to the ATMA framework for the management of multi-hop wireless testbeds using commodity IEEE 802.11 hardware. We believe that the tools described in this section are essential to the management of any multi-hop wireless testbed and therefore should be included in the core set of tools for the management of such testbeds. The tools themselves are based on the classic client-server architecture: the client component of the tool is situated on the ATMA agent and the server component on the ATMA manager.

A. Testbed Configuration Tool

The Testbed Configuration Tool helps control testbed devices. It operates in two phases. In the first phase, the tool discovers the hardware and software configuration of the testbed devices in the following manner. The server component of the tool instructs the client components to send it device configuration details, e.g. the number of interfaces on the device and the hardware specifics. Upon receiving such a request, the client logs into the testbed device, discovers the requested device configuration, and sends the resulting information to the server.

In the second phase, the server instructs the clients to configure the testbed devices with the configuration parameters supplied by the testbed operator.

B. Interference Meter

For the purposes of testbed configuration and performance evaluation, it is useful to characterize the level of interference in the wireless medium. Quantifying the level of interference, however, is a challenging task. Accurate characterization can be achieved using expensive hardware. However, such characterization may not always be feasible because of bulky hardware, the time varying nature of interference, and the large area over which testbed nodes might be distributed.

As a first approximation, however, a characterization of the number of simultaneously operating wireless networks that use the same physical layer technology can be a good estimate of the level of interference[4]. Measuring the number of wireless networks operating on the channel is easily done since most wireless hardware support a channel scanning operation that listens for IEEE management frames (beacons) in the medium. These beacons are wireless network advertisements transmitted by access points operating in IEEE 802.11 infrastructured mode and by devices operating in IEEE 802.11 ad hoc mode. The tool we have developed to measure interference uses the above mentioned channel scanning method to characterize interference. Upon receiving a request from the server component of this tool to sample the various channels, the client does a scan of all channels within a small time period[5]. Once the scan is completed, the client sends the scan results to the server. By co-relating the scan results received from all clients in the testbed, the testbed operator can compute an approximation of the level of interference throughout the testbed.

As an example of how the approximation can be computed, the level of interference on a channel c can be estimated using the metric I_c, given as:

$$I_c = \frac{\sum_{i=1}^{m} N_i^c}{m}$$

where m is the number of testbed nodes and N_i^c is the number of interfering networks as discovered by node i on channel c. The metric I_c can be interpreted as a measure of the average number of networks that interfere with a node in the testbed. A testbed operator can then use this metric to select a channel for the testbed with the least number of interfering wireless networks, therefore minimizing interference between the testbed and co-located wireless networks.

C. Network Monitoring

In prior work, we developed DAMON [15], a distributed network monitoring tool for mobile networks. DAMON relies on a distributed set of agents within the network to collect information and send that information to sinks where it can be stored. DAMON collects information such as the number of packets sent and received in the network, topology data, routing table information at intermediate nodes, and quality of various links in the network. By using DAMON, a testbed operator can obtain valuable information about the state of the testbed. This can help in fault detection/isolation, network troubleshooting, and performance evaluation.

We use DAMON as the monitoring tool in the extension developed for multi-hop wireless testbeds. DAMON, however, was originally designed to transport monitoring information

[3]It is assumed that the ROUTE-REPLY contains information to uniquely identify an agent.

[4]However, characterizing interference caused by devices that use other physical layer technologies, such as cordless phones or BlueTooth devices, operating in the same frequency spectrum cannot be achieved using this technique.

[5]The sampling period is dependent on the hardware used for the scan.

Fig. 2. UCSB Wireless Testbed Device.

in-band. In-band operation has two disadvantages: (1) the delivery of monitoring information is overhead and can therefore lead to an inaccurate assessment of network performance; and more importantly, (2) an outage in the network can result in the failure of the delivery of monitoring information required to analyze and troubleshoot the outage itself.

Since information can be delivered over the ATMA mesh, we have modified DAMON to transport collected information over the ATMA mesh instead of delivering it in-band. This reduces overhead in the testbed network and also ensures that information required for an analysis of the testbed is always available regardless of the actual state of the testbed.

D. Topology Control Tool

In experimenting with protocols and systems, it is necessary to test the protocols and systems in different network topologies. Creating different topologies through the movement of testbed nodes is a time-consuming, labor-intensive operation. Therefore, we have developed a topology control tool that can create virtual topologies without physically moving nodes. While not all desired topology configurations can be constructed virtually, virtual topologies are effective at creating different configurations and environments for testing. We briefly describe the operation of this tool below.

Given an initial topology, the tool creates the virtual topology by selectively dropping packets from nodes in the network that are not present in the virtual topology. The selective dropping of packets is done in the following manner. The server component of the tool uses the testbed topology and the virtual topology to determine which nodes in the network need to be masked. Given a testbed node to be masked, the server instructs all clients who are neighbors of the masked node to filter all packets received from that node at their respective testbed nodes. The actual filtering of packets is achieved using operating system primitives. For example, in Linux the netfilter framework can selectively filter packets based on certain criteria such as IP address and port number.

V. Implementation

We have implemented the ATMA framework using the Linksys WRT54G wireless router. The WRT54G is a wireless device capable of operating in IEEE 802.11b/g mode. It supports the Linux operating system and is capable of operating in both infrastructured and ad hoc modes of operation specified in the IEEE 802.11 standard.

Figure 2 illustrates two views of a ATMA mesh node connected to a testbed device deployed as part of a twenty-five node multi-hop wireless testbed in UC Santa Barbara. The testbed device is also a WRT54G device and operates in ad hoc mode. The testbed device and the ATMA node, however, communicate on different channels to minimize interference. The ATMA node uses the OpenWRT Linux distribution [2] as its operating system and a modified version of AODV [14] for routing within the mesh. Our modified AODV implementation discovers multiple node-disjoint paths to a destination and selects a path using a reliability-based metric, instead of the shortest hop metric as specified in the protocol standard. The reliability of a path is computed as the product of the reliabilities of each link in the path. The reliability of a link is calculated by sending sequence-numbered unicast probes in both directions of the link. We chose the reliability-based metric for two reasons: (1) a shortest hop count metric has been shown to deliver traffic poorly [5]; and (2) a reliability-based metric will ensure that management commands sent

on the ATMA mesh are delivered using the most reliable route. In addition to supporting the reliability-based metric, we overloaded the function of the AODV HELLO control message to carry manager beacons.

The ATMA manager, ATMA agent, and all tools described in Section IV are implemented using the C programming language and shell scripting. All ATMA control messages are carried using the UDP transport protocol to minimize the overhead of connection setup and tear-down associated with TCP. In Section III-C, we described the ATMA agent initialization process where the agent randomly picks a temporary IP address to register with the manager. In choosing the IP address, the agent uses its network interface address as the random seed.

VI. Conclusions

Conducting experiments in scenarios that closely resemble a target deployment is challenging because of the lack of means to access testbed nodes out-of-band. As a result, testbeds are typically deployed where such access is feasible. Consequently, protocols and systems designed using such testbeds do not perform as predicted when used in a target deployment.

In this paper, we presented the ATMA framework for wireless testbeds which can be installed in regions where deployment was not previously possible because of accessibility concerns. The ATMA framework relies on a multi-hop mesh network to provide a testbed operator with an out-of-band technique to manage the testbed. The ATMA mesh is designed to be self-configuring and therefore requires minimal configuration when deployed. As an extension of the ATMA framework to multi-hop wireless testbeds, we have developed a suite of tools designed for the management and monitoring of such testbeds.

As part of our future work, we plan to continue development of the ATMA framework and to offer it for download for other researchers to use in their testbeds.

References

[1] AODV@IETF project. http://moment.cs.ucsb.edu/aodv-ietf/.

[2] OpenWRT project. http://www.openwrt.org/.

[3] B. Chambers. The Grid Roofnet: A Rooftop Ad Hoc Wireless Network. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 2002.

[4] S. Cheshire, B. Aboba, and E. Guttman. Dynamic Configuration of IPv4 Link-Local Addresses. Internet Engineering Task Force (IETF), draft-ietf-zeroconf-ipv4-linklocal.txt, July 2004.

[5] D. De Couto, D. Aguayo, B. A. Chambers, and R. Morris. Performance of Multihop Wireless Networks: Shortest Path is Not Enough. In *Workshop on Hot Topics in Networks*, Princeton, NJ, October 2002.

[6] R. Draves, J. Padhye, and B. Zill. Comparision of Routing Metrics for Static Multi-hop Wireless Networks. In *ACM Sigcomm*, Portland, OR, August 2004.

[7] R. Draves, J. Padhye, and B. Zill. Routing in Multi-radio, Multi-hop Wireless Mesh Networks. In *ACM International Conference on Mobile Computing and Networking*, Philadelphia, PA, September 2004.

[8] R. Gray, D. Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Masone, S. McGrath, and Y. Yuan. Outdoor Experimental Comparision of Four Ad Hoc Routing Algorithms. In *ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Venice, Italy, October 2004.

[9] L. Gruenwalk, M. Javed, and M. Gu. Energy-Efficient Data Broadcasting in Mobile Ad-Hoc Networks. In *International Symposium on Database Engineering and Applications*, Edmonton, Canada, July 2002.

[10] H. Lundgren, D. Lundberg, E. Nordstrom, C. Tschudin. A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations. In *IEEE Wireless Communication and Networking Conference*, Orlando, FL, March 2002.

[11] C. Hsin and M. Liu. A Distributed Monitoring Mechanism for Wireless Sensor Networks. In *ACM Workshop on Wireless Security*, Atlanta, GA, 2002.

[12] K. Fall and E. Varadhan. NS Notes and Documentation. In *http://www-mash.cs.berkeley.edu/ns/*, 1999.

[13] D. Kotz, C. Newport, R. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental Evaluation of Wireless Simulation Assumptions. In *ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Venice, Italy, October 2004.

[14] C. Perkins, E. Belding-Royer, and S. Das. Ad Hoc On-Demand Distance Vector Routing. Internet Engineering Task Force (IETF), RFC 3561, July 2003.

[15] K. Ramachandran, E. Belding-Royer, and K. Almeroth. DAMON: A Distributed Architecture for Monitoring Multi-hop Mobile Networks. In *IEEE International Conference on Sensor and Ad hoc Communications and Networks*, Santa Clara, CA, October 2004.

[16] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols. In *Wireless Communications and Networking Conference*, New Orleans, LA, March 2005.

[17] Y. Tseng, S. Ni, Y. Chen, and J. Sheu. The Broadcast Storm Problem in Mobile Ad Hoc Networks. In *ACM International Conference on Mobile Computing and Networking*, Seattle, WA, August 1999.

[18] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks. In *Workshop on Parallel and Distributed Simulations*, Banff, Canada, May 1998.

[19] J. Zhao and R. Govindan. Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. In *International Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, November 2003.

[20] J. Zhao, R. Govindan, and D. Estrin. Computing Aggregates for Monitoring Wireless Sensor Networks. In *International Workshop on Sensor Net Protocols and Applications*, San Diego, CA, April 2003.