

Resource Monitoring Issues in Ad Hoc Networks

Cristian Tuduce

Thomas Gross

Departement Informatik

ETH Zurich

8092 Zurich, Switzerland

Abstract—In this paper we explore the solution space for resource monitoring systems for mobile ad hoc networks. To cope with the range of parameters of interest to an application, we separate the resource monitoring system into an extensible infrastructure and specific sensors, which record network parameters of interest. The monitoring infrastructure includes two abstractions that capture a node’s environment, the neighborhood and the swath. Whereas in a wire-line network information about the nodes along the path from source to destination is sufficient to characterize the connection between source and destination, in mobile networks other nodes within transmit range may have an impact. The paper concludes with a discussion of our experience obtained with a prototype implementation and a summary of preliminary results.

I. INTRODUCTION

MANY networked applications include provisions to adapt the behavior or functionality to best use the available computing and networking resources. These applications use one or more environment parameters to perform their adaptation. Adaptive applications are already recognized as important in the world of (managed) local and wide area networks. Because of the broader range of environment parameters that is encountered in mobile ad hoc networks, even more adaptation is expected from applications as they enter the ad hoc world.

There are two fundamental problems that must be addressed: (1) an application must obtain information about its environment, and (2) the application must react to these changes. We focus here on the first issue since it must be addressed before any adaptation is possible. Furthermore, some adaptation responses may be highly application-specific. Adaptation on a mobile node can happen at many levels: the (end-user) application may adapt, the middleware system may adapt, or the protocols may adapt. Since we focus on the system that provides the resource information, we postpone a discussion of this aspect; we use the term *client* to refer to the part of the system that consumes the information about resource availability.

A resource monitoring system eases the design and development of adaptive software systems by decoupling resource information gathering from decision making. In the past years, a number of resource monitoring systems have been described and implemented for conventional (managed) networks [1], [2], [3], [4], [5]. The deployment of these monitoring systems

in ad hoc networks faces problems. First, it is safe for monitoring systems for conventional networks to rely on services offered by the network infrastructure (e.g., Remos [3] expects routers or access points with SNMP support). This expectation is however unrealistic for ad hoc networks, where the nodes form the networking fabric, and the functionality that all nodes must provide must be kept as simple as possible. Second, the architecture of the current monitoring systems is centralized with respect to information modeling, i.e. the transformation of raw sensor data into metrics that are meaningful to the clients of the monitoring service. Although resource monitoring systems are well developed for conventional networks, they are still an open issue in mobile ad hoc networks.

There are many usage scenarios for resource monitoring systems in ad hoc networks e.g., multimedia streaming adaptation [6] and quality of service (QoS) routing, a topic that has received considerable attention by a number of researchers [7], [8], [9], [10], [11]. QoS routing systems use resource monitoring for admission control and for QoS contract enforcement.

Previous experience with resource monitoring systems shows that exposing all network parameters to all clients (applications) is not a good idea. It is better to leave open details of the kind of information an application inquires. Because of the large number of environment parameters and their characteristics encountered in ad hoc networks, we split the monitoring systems architecture into a common infrastructure and a resource-specific part. This organization (also chosen for other monitoring systems) decouples the resource monitoring infrastructure from the modules that gather the information about resource(s). The interface between client and monitoring system is query based so that the client can control how much interaction is needed. Finally, the responsibility to keep track of histories rests with the modules that gather environment parameter values; each module decides if (and how) a history should be maintained.

The rest of the paper presents the design and implementation of a resource monitoring and reporting infrastructure that considers the properties of ad hoc networks. We discuss the challenges raised by this type of networks and trade-off possible solutions. We report on preliminary results and experience with the prototype implementation in an ad hoc network testbed.

A. The Network Model

An ad hoc network is formed by computing or sensor nodes with communication capabilities. In the prototype system, nodes communicate via 801.11-compliant air interfaces, but

This work was funded, in part, by the NCCR “Mobile Information and Communication Systems”, a research program of the Swiss National Science Foundation, and by a gift from Intel’s Microprocessor Research Laboratory.

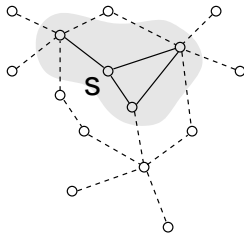


Fig. 1. Neighborhood abstraction example. Highlighted is the direct neighborhood of node s .

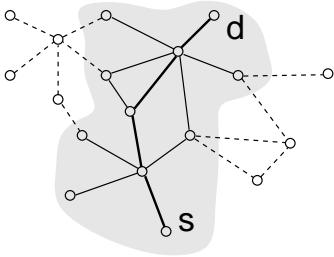


Fig. 2. Swath abstraction example. Highlighted is the swath from node s to the destination node d with a width of 1.

the exact communication technology is of secondary importance. The crucial aspect is that the nodes are able to opportunistically build a network among them. There is a network link between two nodes if they are in communication range of each other. If two nodes are not within communication range, they can communicate using other nodes as packet relays. In the network model considered in this paper, the links between nodes are bidirectional. Thus, if node n_i can transmit a packet to node n_j , node n_j can transmit a packet to n_i . The network can be regarded as a graph, where the set of vertices is the set of mobile nodes and the set of edges is the set of link between the nodes.

Since ad hoc networks conforming to this model lack the existence of specialized routers or switches, all the monitored resources are located at the networked nodes. The problem of monitoring the resources at a set of nodes can be reduced to periodically selecting the sub-graph containing the specified set of nodes. The sub-graph is selected by criterions specified by the application and represents the application's view of the environment. The vertices in the selected sub-graph store information about each node's resources (e.g., network interface usage, transmit queue length, transmit power, and/or battery level if the node is operating on batteries). The set of edges reflects the network topology.

II. NETWORK ABSTRACTIONS

One of the challenges of designing a monitoring infrastructure for ad hoc networks is to find abstractions that capture the properties of ad hoc networks yet are general enough to apply to the (unconstrained) topologies that can be realized by such networks. The central aspect of mobile ad hoc networks is that seemingly unrelated nodes can have a (negative) influence on an application. Two abstractions cover this simplified view of the network:

- **Neighborhood** – This abstraction offers the client a view of the environment around a node. An example of the neighborhood abstraction is depicted in Figure 1. In this example, the client requests the direct neighborhood of node s .
- **Swath** – This abstraction offers the client a view of the environment along a path. Because in wireless networks a node's transmission is influenced by the transmission of other nodes, we want to allow the client to specify the width of the swath. The client can therefore capture in its view the nodes neighboring the path. An example of the swath abstraction is depicted in Figure 2. In this example the client requests the swath from the source node s to the destination node d . The swath abstraction can be used to monitor resources that have a node-to-node semantic, say, bandwidth monitoring from node s to node d .

III. MONITORING INFRASTRUCTURE DESIGN

The common infrastructure of the monitoring system must provide the interface to clients and it must work in a distributed fashion. The monitoring infrastructure offers clients a query-based interface. The clients issue queries to the monitoring system and the monitoring system replies with the requested view of the environment. We assume that all nodes in the network participate in the resource monitoring process. A client running on a mobile node contacts the local monitoring module (MM). To build the answer to the query, the MM collaborates with MMs on other nodes.

We allow for two types of queries, corresponding to the two abstractions: neighborhood queries and swath queries. Based on these basic queries, higher layer queries can be specified, e.g., the energy map of the network, or the path with the highest network quality parameters.

The reply to a query is a graph $G(V, E)$, where V is a set of vertices and E is a set of edges. The vertices in the set correspond to the nodes in the network. There is an edge between two vertices if the two corresponding nodes are within communication range.

A. The Neighborhood Query

The answer to a neighborhood query is the topological graph G of the neighborhood of the querying node. The querying node specifies the maximum radius R of the graph returned by the monitoring system. The radius is measured in nodes (or network hops). In Figure 1, the shaded area is the reply to a neighborhood query issued at node s with the radius $R = 1$. The query is disseminated in an expanding ring, until the desired radius of the query is reached. The answers are sent to the nodes from which the query was received. If an intermediate node receives answers from multiple nodes, it merges the received answers with its own answer to the query. Duplicate entries are suppressed.

B. The Swath Query

The parameters of this type of queries are the destination node n_d and the width of the swath W . The answer to this

query is the topological graph G of the neighborhood of the path from n_s to n_d . Node n_s is always the node where the query originated. The width of the graph is expressed in nodes. The width of the swath is calculated from the path between n_s and n_d to the edge of the swath. If we consider the network depicted in Figure 2, the shaded area is the answer to a swath query issued at node s for the destination node d with the specified width of 1 hop.

The swath query is disseminated first along the path from node n_s to node n_d , then to the neighbors of the path, until the width requirement is satisfied.

All queries return the status quo of the network, at the time information was gathered. It is possible that that information is out of date by the time it reaches the requesting node. Since only the client knows the sensitivity of its decision making to stale data, we leave it to the client how to react.

IV. NODE CLUSTERING

Theoretical analysis [12], as well as experimental studies [13], show that in ad hoc networks only a fraction of the peak bandwidth is available to applications, the rest of the bandwidth is used by the management traffic. Therefore, every new management protocol or system should take a conservative approach with regards to bandwidth consumption. The goal of information accuracy is however conflicting with the goal of minimal intrusion. The bandwidth consumed by the monitoring system can be reduced by reducing node mobility or by reducing the number of monitoring updates. We will focus on solutions to lower the number of management packets. One possible solution to reducing the monitoring updates is node clustering. Several clustering algorithms for mobile ad hoc networks have been proposed in the past years [14], [15], [16], [17], [18]. The aforementioned algorithms are approaching a similar problem: lowering the management packets for routing in mobile ad hoc networks.

The clustering algorithm to group the mobile nodes is derived from a clustering protocol due to Gerla et al. [18]. This algorithm is based on the *first declaration wins* (FDW) rule. When a node powers on, it becomes cluster-head if it has no neighbors in the cluster-head state. A node that is assigned the cluster-head role remains cluster-head until due to mobility, the cluster-head moves in direct communication range with another cluster-head. This solution proved to provide more stable clusters than the weight based clustering algorithms [16], [14]. Weight based algorithms are recomputing the state of the nodes every time the topology of the network changes. Furthermore, the convergence time of the FDW algorithm is lower than that of weight based clustering algorithms.

The FDW algorithm described in [18] implements passive clustering. This means no management packets are used for the clustering protocol. The clustering information is inserted in the MAC header of the data packets. A node can be in one of these five states: INITIAL, ORDINARY_NODE, GATEWAY, CLUSTERHEAD-READY and CLUSTERHEAD. Nodes in CLUSTERHEAD-READY and INITIAL states do not belong to any cluster. These nodes wait for data packets to be received, to assess the next state of the node.

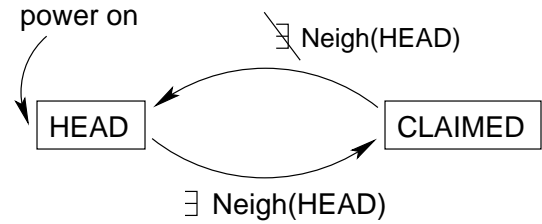


Fig. 3. The state machine of the clustering protocol.

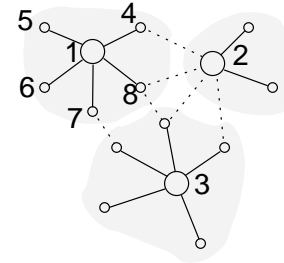


Fig. 4. Node clustering and role distribution example.

A disadvantage of this protocol is that nodes can be in states where they do not belong to any cluster. In the resource monitoring system, all the nodes must be at all times under monitoring control. To fit this algorithm to the needs of a monitoring system, we eliminate the states where the nodes are not associated with any cluster. This elimination comes however at the cost of introducing cluster management packets.

One node per cluster is assigned the cluster-head role. The other nodes in the cluster, the so called *claimed nodes*, report to the cluster-head. The state machine of the modified algorithm is depicted in Figure 3. A node in the HEAD state is a cluster-head. Nodes in the CLAIMED state are always associated with a cluster-head.

Figure 4 depicts a node clustering example. The shaded areas represent the clusters. The big circles are cluster-head nodes (nodes 1, 2 and 3). The small circles are claimed nodes (e.g., the claimed nodes of 1 are nodes 4, 5, 6, 7 and 8).

Each cluster-head maintains an up to date view of its cluster. Consider the network depicted in Figure 4. Figure 5 depicts the graph maintained at cluster-head 1. The solid lines represent edges inside the cluster. The dashed lines represent *fringes*. A fringe is a link to a node that does not belong to the current cluster. Fringes are used to express connectivity between clusters, to express the fact that the network continues beyond the current view of the network.

A claimed node sends its local information to the cluster-head it is associated with. The local information maintained at each node depends on the number of information gathering

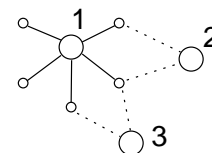


Fig. 5. Building graphs at the cluster-heads from the physical layout of the network. Topology information maintained at cluster-head 1.

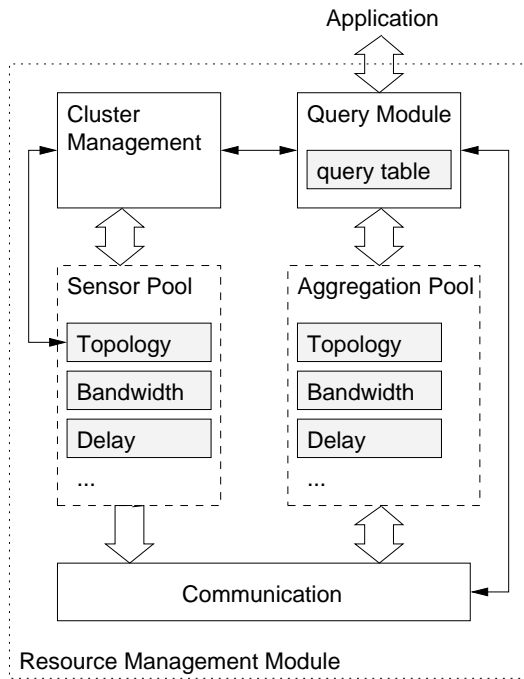


Fig. 6. The architecture of the resource management module.

modules deployed on the mobile node. Topology gathering is a prerequisite for the clustering algorithm. Therefore, the topology gathering module is included in the common monitoring infrastructure and is always present at any node. Topology information updates are sent just when topology changes occur. Resource usage updates are sent regularly every t_{update} seconds.

The cluster-head gathers the information from its claimed nodes and answers to client queries on behalf of the cluster members. Queries are disseminated only to cluster-heads, and only cluster-heads are allowed to answer queries. The node at which the client is issuing the query can be a claimed node or a cluster-head. If the node is a claimed node, it forwards the query to the cluster-head. The monitoring system disseminates queries and builds the reply to queries as described in Section III.

V. THE RESOURCE MONITORING MODULE ARCHITECTURE

The resource monitoring module (RMM) is the unit of the resource monitoring system running on every mobile node. Figure 6 depicts the internal architecture of the monitoring module. A RMM contains five main modules: the communication module, the cluster management, the query module, the sensor pool, and the aggregation pool. These modules implement the common monitoring infrastructure. The parameter specific modules are loaded into the sensor pool and into the aggregation pool. The aggregation pool is the place that holds the modules that process, i.e., transform the raw data obtained by the various sensors.

The communication sub-module is an interface to the operating system's network communication fabric. Data exchanged by RMMs is serialized by this sub-module.

The sensor pool is an open list of environment parameter sensors. The sensor sub-modules gather local information at the node on which the monitoring system is running. The only required sensor is the topology sensor. The topology sub-modules stores neighbor information. The neighbor information contains the address of each neighbor and the cluster ID of the cluster to which the neighbor belongs. If the RMM is in CLAIMED state, the sensors in the sensor pool send resource usage updates to the cluster-head.

At the cluster-head, resource usage updates are received by sub-modules in the aggregation pool. These sub-modules aggregate resource usage information received from the sensor pool of the claimed nodes. In addition, the sub-modules also join information received from other cluster-heads as replies to queries. Each sub-module in the aggregation pool must have its equivalent sub-module in the claimed node's sensor pool. The sensor pool and aggregation pool allow for later development and deployment of environment sensing modules.

The cluster management sub-module implements the state machine of the clustering protocol. Based on the neighbor information received from the topology sensor, this sub-module assesses the role of the RMM on this host as described in Section IV. The cluster management sub-module periodically broadcasts beacons. The information disseminated in the beacon contains the address of the host, the state of the cluster management sub-module and the cluster ID to which this node belongs. The beacons are received by the topology sub-modules of the neighboring nodes. Beacons are transmitted every t_{beacon} seconds.

The query sub-module implements the interface to clients. This sub-module can operate in two modes: forwarding mode and answering mode. The query sub-module is in forwarding mode when the RMM is in CLAIMED state. In this state client queries are simply forwarded to the cluster-head node. In this mode the host does not store history information for other nodes. The answering mode is used when the host is in HEAD state. In this mode, information about the resource usage for the entire cluster is stored in the aggregation pool sub-modules.

When the answer to client queries cannot be generated locally, the `query_table` structure stores information about the in-progress queries. Each in-progress query has an associated timer. If the answer to the query is not built in time (e.g., due to a cluster-head node failure), after the reply time-out a partial query answer is returned to the client, and the reply is marked as incomplete.

VI. EVALUATION

We implemented a prototype of the resource monitoring system for the Linux OS. We made a preliminary evaluation of the system to assess the way the monitoring system copes with the challenges raised by ad hoc networks.

The evaluation was conducted on an ad hoc network test-bed of 13 nodes. The participating nodes are laptop computers running the Linux operating system. Nine of them are Dell Latitude laptops with an Intel Pentium 4 CPU at 1988 MHz. The other four are IBM ThinkPad 560X systems with a Pentium CPU at 233 MHz. Networking was enabled by

TABLE I
NEIGHBORHOOD QUERY ANSWER TIME [MS] FOR DIFFERENT QUERY
SIZES.

Radius [hops]	@ Head		@ Claimed	
	min.	max.	min.	max.
0	0	0	12	287
1	11	110	234	298
2	91	249	233	309
3	54	383	116	976
4	98	389	372	851
5	254	773	315	1143
6	393	760	398	1012

wireless LAN (IEEE 802.11b) cards in ad hoc mode. We used the AODV routing algorithm to enable multi-hop connectivity between nodes. The *kernel-aodv1.2* implementation of AODV was used. To emulate node mobility and induce different connectivity patterns we used the *MobiEmu* [19] package.

The resource monitoring module is implemented as a daemon in the user-space of the operating system.

In the first test, we measure the agility of the clustering algorithm. The agility influences the capability of the system to adapt to mobility. The lower the time in which the cluster reorganizes, the more mobile the network can be. The cluster reorganization time is influenced by two parameters: the beacon time and the neighbor active time. Each beacon time interval a node sends a beacon with cluster information about itself (e.g., state, association). The neighbor active time is the time after which not receiving a packet from a neighbor we consider the connectivity with the neighbor lost.

For this test we conducted ten measurements, the figures are minimum and maximum reorganization time over these ten measurements. The beacon time is 2s, the neighbor active time is 5s.

When a node powers on in the middle of an active network, it takes 3-10ms for the node to be integrated in the cluster structure. When active nodes move towards each-other, it takes between 1200ms and 1358ms to detect the new neighbors and reorganize the clusters. When nodes move away from each-other, it takes 4250-4938ms to detect the departure and to reorganize the clusters. This delay is mainly influenced by the value of the neighbor active timer.

In the second test we measured the time it takes for the neighborhood query to be answered. We measured the query reply time experienced by a client. We varied the radius of the query from 0 to 6 hops. The query reply times are shown in Table I. The figures in the table are minimum/maximum over ten measurements. For queries issued at cluster-head nodes, the cluster structure is reorganized between 0 and 773ms. When the queries are issued at the claimed nodes, the cluster reorganizes in 12-1143ms.

We measure the answer time also for swath queries. We keep the width of the swath constant and vary the query length. The swath width is 0 nodes. This means the reply to the query contains just the path from the source to the destination, without additional path neighbors. The time it takes the monitoring system to answer swath queries is presented in Table II. At the cluster-head nodes, swath queries are answered

TABLE II
SWATH QUERY ANSWER TIME [MS] FOR DIFFERENT QUERY SIZES.

Length [hops]	@ Head		@ Claimed	
	min.	max.	min.	max.
1	33	808	42	1004
2	158	411	236	545
3	243	523	301	597
4	394	880	528	871
5	567	1672	631	1944
6	680	1377	713	1604

between 33 and 1672ms. For queries issued at claimed nodes, the system answers in 42-1944ms.

From the measurements it is evident that the monitoring system is agile enough to run in a mobile ad hoc network.

VII. RELATED WORK

For local and wide area networks, Remos [3] (Resource Monitoring System) and the Network Weather Service [2] offer adaptive applications information about the state of the network in which they are running.

Remos [3] allows network-aware applications to obtain relevant information about their execution environment. Remos consists of a modeler and one or multiple collectors. Collectors gather information about the network. The modeler combines data gathered from collectors and implements the API for the applications. The Remos system does not implement any collector targeted specifically at information gathering in mobile ad hoc networks. Parts of the monitoring system described in this paper could be wrapped to act as a Remos collector.

The Network Weather Service (NWS) [2] provides forecasts of performance characteristics from a distributed set of metacomputing resources. The NWS consists of a network of sensors that monitor the performance of networks and processors. The measurements from the sensors are processed at a forecaster module. This module provides forecasts for different performance characteristics and implements the interface to the applications. Neither does NWS implement sensors for performance characteristics of ad hoc networks, nor is the forecasting module generating forecasts related to mobile ad hoc network specifics.

Like the monitoring system described in this paper, network management protocols gather information about the state of the network. However, their primary task is to react with management actions to network state changes and not to offer applications a view of the environment in which they are running. The Ad hoc network Network Management Protocol (ANMP) [20] is a hierarchical approach to network management. The nodes in the networks are organized in clusters. The cluster-heads are polling their cluster members in a centralized fashion. The clustering used in this protocol is based on sampling periods. Clusters are reorganized at discreet time intervals. If a node moves between the sampling intervals, the new cluster will detect the presence of the newly arrived node just at the next sampling interval. This way, moving nodes are not managed between sampling intervals.

The Guerrilla Management Architecture (GMA) [21] brings continuity in ad hoc network management. As well as in ANMP, in GMA the nodes are organized in clusters. Nodes can have different management tasks based on their capabilities. The nodes form a management hierarchy.

VIII. CONCLUDING REMARKS

In this paper we present a resource monitoring architecture for mobile ad hoc networks. The system is based on the abstractions of neighborhood and swath. These abstractions capture the set of nodes that may have an influence on a given node's communication operations.

The resource monitoring system presented here is structured into a common infrastructure for resource monitoring that leaves the details of performance measurement and processing to the clients of the resource information. This approach decouples applications (and other users of resource information) from the details of the information gathering.

The system is implemented and has been evaluated in a small-scale ad hoc network. Preliminary experience indicates that the monitoring system is agile enough to run in a highly mobile ad hoc network. As mobile ad hoc networks become a platform for applications, these applications will need information about the network's resource situation. The resource monitoring system described here provides a simple yet effective solution to a set of problems faced by all ad hoc networks.

REFERENCES

- [1] C. A. Lee, J. Stepanek, R. Wolski, C. Kesselman, and I. Foster, "A Network Performance Tool for Grid Environments," in *Proc. 11th IEEE Symp. High-Perf. Dist. Comp. (HPDC)*, pp. 260–67, Jul. 1998.
- [2] R. Wolski, N. Spring, and J. Hayes, "The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing," *Journal Future Gen. Comp. Sys.*, vol. 15, pp. 757–68, Oct. 1999.
- [3] P. Dinda, T. Gross, R. Karrer, B. Lowekamp, N. Miller, P. Steenkiste, and D. Sutherland, "The Architecture of the Remos System," in *Proc. 10th IEEE Symp. on High-Perf. Dist. Comp. (HPDC'01)*, (San Francisco), pp. 252–65, Aug. 2001.
- [4] C. Liu, L. Yang, I. Foster, and D. Angulo, "Design and Evaluation of a Resource Selection Framework for Grid Applications," in *Proc. 11th IEEE Symp. High-Perf. Dist. Comp. (HPDC)*, pp. 63–72, Jul. 2002.
- [5] D. Garlan, D. Siewiorek, A. Smilagic, and P. Steenkiste, "Project aura: Toward distraction-free pervasive computing," *IEEE Pervasive Computing*, pp. 22–31, April-June 2002.
- [6] C. R. Lin and M. Gerla, "Asynchronous multimedia multihop wireless networks," in *Proc. IEEE INFOCOM*, pp. 118–25, Apr. 1997.
- [7] C. Zhu and M. S. Corson, "QoS routing for mobile ad hoc networks," in *Proc. IEEE INFOCOM*, vol. 2, pp. 958–67, Jun. 2002.
- [8] T.-W. Chen, J. T.-C. Tsai, and M. Gerla, "QoS Routing Performance in Multihop, Multimedia, Wireless Networks," in *Proc. of IEEE ICUPC*, vol. 2, pp. 557–61, 1997.
- [9] Y.-C. Hsu and T.-C. Tsai, "Bandwidth Routing in Multihop Packet Radio Environment," in *Proc. of 3rd Int. Mobile Comp. Workshop*, Mar. 1997.
- [10] S. Chen and K. Nahrstedt, "Distributed Quality-of-Service Routing in Ad-Hoc Networks," *IEEE Journal Sel. Areas Comm.*, vol. 17, pp. 1488–505, Aug. 1999.
- [11] E. M. Belding-Royer, C. E. Perkins, and S. R. Das, "Quality of Service for Ad Hoc On-Demand Distance Vector Routing." draft-ietf-manet-aodvqos-00.txt, Internet Draft, IETF, Jul. 2000. (Work in progress).
- [12] P. Gupta and P. R. Kumar, "The Capacity of Wireless Networks," in *IEEE Trans. Inf. Theory*, vol. 46, pp. 388–404, March 2000.
- [13] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris, "Capacity of Ad Hoc Wireless Networks," in *Mobile Comp. Net.*, pp. 61–69, 2001.
- [14] C. R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal Sel. Areas Comm.*, vol. 15, pp. 1265–75, Sep. 1997.
- [15] S. Basagni, D. Turgut, and S. K. Das, "Mobility-Adaptive Protocols for Managing Large Ad Hoc Networks," in *Proc. Intl. Conf. on Comm.*, vol. 5, pp. 1539–43, June 2001.
- [16] M. Chatterjee, S. K. Das, and D. Turgut, "WCA: A weighted clustering algorithm for mobile ad hoc networks," *Cluster-Computing 5*, pp. 193–204, 2002.
- [17] P. Sinha, R. Sivakumar, and V. Bharghavan, "Enhancing Ad Hoc Routing with Dynamic Virtual Infrastructures," in *Proc. IEEE INFOCOM 2001.*, vol. 3, pp. 1763–72, Apr. 2001.
- [18] M. Gerla, T. J. Kwon, and G. Pei, "On-demand routing in large ad hoc wireless networks with passive clustering," in *Proc. Wireless Comm. and Networking Conf.*, vol. 1, pp. 23–28, Sept. 2000.
- [19] Y. Zhang and W. Li, "An Integrated Environment for Testing Mobile Ad-Hoc Networks," in *Proc. Int. Symp. Mobile Ad Hoc Net. Comp. (MOBIHOC)*, (Lausanne), pp. 104–11, Jun. 2002.
- [20] W. Chen, N. Jain, and S. Singh, "ANMP: Ad hoc network Network Management Protocol," *IEEE Journal Sel. Areas Comm.*, vol. 17, pp. 1506–31, Aug. 1999.
- [21] C.-C. Shen, C. Jaikao, C. Srisathapornphat, and Z. Huang, "An adaptive management architecture for ad hoc networks," *IEEE Comm. Mag.*, vol. 41, pp. 108–15, Feb. 2003.