

Adaptive Quality of Service for a Mobile Ad Hoc Network

Antonis Dimakis, Linhai He, John Musacchio, Hoi-Sheung Wilson So, Teresa Tung, Jean Walrand
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720
{dimakis, linhai, musacchj, so, teresat, wlr}@eecs.berkeley.edu

Abstract—This paper presents a QoS routing system for MANET supporting multiple traffic classes. The system takes into consideration clustering and channel allocation. Simulation experiments show that our algorithms are convergent. The system also yields a higher total throughput compared to the case in which every interface uses the same channel.

I. INTRODUCTION

A. Motivating Scenario

Cellular phone systems have been tremendously successful but they rely on a fixed infrastructure. In some cases, such as after an earthquake, it is desirable to set up a network in an area without any infrastructure as quickly as possible. In this paper, we present the design and simulation results of a QoS routing system that can provide different classes of service in a mobile ad hoc network. We believe the entire QoS routing system should be considered as a whole because optimizing different components of a system separately may lead to poor performance, or worse, an unstable system. In this paper, we take a first step towards an integrated system by considering clustering and channel allocation in addition to routing.

B. Related Work

Traditionally, QoS routing applies to only fixed wireline networks, see e.g., [2] and the references therein. There have also been many ad hoc routing protocols proposed for MANETs without QoS support as in [12]. Recently, there have been proposals to provide QoS in MANETs as in [3], [9]. In [3] the authors propose an algorithm to minimize utilization that does not take into account the effects of the coupling of interference between neighboring nodes. The work in [10] considers coupling, but the authors use a centralized integer programming approach that does not scale well as the number of nodes becomes large.

Previous work on clustering focuses on metrics such as cluster diameter ([8],[1]) and stability [11]. Our work tries to respect the natural clusters by focusing on the number of inter-cluster links. Although such links are necessary for intercluster communication, only a subset of them are used as gateway links. By minimizing the number of intercluster links, one also minimizes the number of unused links. This metric is considered in [6] using a centralized solution for a stationary network. We propose a distributed algorithm designed for a dynamic environment.

The frequency assignment problem for cellular networks has been studied extensively. [5] provides a comprehensive survey on different approaches to this problem. However, those solutions are not suitable for our purpose as they are designed for fixed infrastructure networks. Furthermore, our problem requires each node maintain connectivity to the rest of the network in addition to reducing interference. We solve the channel allocation problem using simulated annealing which has been widely applied to many combinatorial optimization problems.

[14] provides a good coverage of its theory and some of the typical applications. However, most simulated annealing work assumes a central agent has all the state information and fully controls the adaptation process. Our work applies simulated annealing technique in a completely distributed way.

C. Assumptions

In this paper, we will use the terms *node* and *router* interchangeably. We assume each node of the ad hoc network is mobile and has one or more wireless network interfaces. Each interface can operate at one of the many independent channels (i.e., frequencies) chosen from a predetermined set. Equipping nodes with multiple interfaces can increase the capacity of the network. These interfaces can have different maximum transmission ranges and link rates. The rate of a link between two nodes depends on the channel quality. We assume that the interfaces use a contention-based MAC protocol suitable for a multi-hop ad hoc network but the design of such MAC protocol is outside the scope of this paper.

The network supports unicast applications with different priorities and bandwidth requirements. Our system supports the following classes of application in decreasing order of priority: (1) Urgent Messages (UMs) are connectionless and they demand minimum end-to-end delays. (2) Urgent Flows (UFs) are urgent calls with low to medium bandwidth requirements of up to about 100Kbps. They require fast call set-up, fast rerouting in case of failures, but are relatively infrequent. (3) Regular Flows (RFs) are non-urgent voice and video calls with potentially high bandwidth requirements. They are frequent and demand a low blocking probability. (4) Best Effort (BE) flows are connectionless and do not require any bandwidth guarantees.

II. SYSTEM DESCRIPTION

A. System Architecture

Our system consists of three major components: clustering, channel allocation, and route computation. An efficient routing system must support load balancing of traffic. This capability requires some knowledge of the network topology beyond the local neighbors. However, without any hierarchical structure, each node would have to gather global topology information and incur a high control overhead. Our system partitions the network into clusters of nearby nodes and perform hierarchical routing at two levels (within a cluster and across clusters) to reduce control overhead. Another advantage is when a link fails in a transit cluster, local rerouting of flows within the cluster may suffice to contain the fault without bothering the end nodes. Although hierarchical routing is sub-optimal because some links are not used, we believe the differences are small and acceptable.

The clustering algorithm dynamically organizes the nodes

into a two-level hierarchy. Each cluster is connected to nearby clusters via one or more *gateways* which are nodes at the boundary of two clusters. These gateways, also determined by the clustering algorithm, are responsible for finding routes for flows crossing clusters. The goal of the channel allocation algorithm is to assign, in a distributed fashion, suitable channels to different interfaces to maximize the network throughput while keeping the network connected. The channel allocation algorithm implicitly defines the links in the network.

Within each cluster, each node participates in a link state protocol to exchange information on the topology within the cluster. In addition, all gateways in the network participate in another link state protocol exchanging connectivity information at the cluster level. Based on the link information within a cluster and the cluster summaries exchanged among the gateways, a distributed routing algorithm finds the best routes for different flows and traffic classes. The QoS routing algorithm is also responsible for rerouting flows after a failure. We describe each of these three components in the following sections in details.

B. Clustering Algorithm

Our objective is to form clusters near size n while minimizing the resulting number of intercluster links. We choose the target cluster size n to be the square root of the total number of nodes in order to minimize the sizes of the routing tables. In our clustering algorithm, each node maintains two types of clustering: prospective and actual. The following distributed algorithm computes the prospective clustering. A leader node is associated with each prospective cluster. Initially, every node starts as its own prospective cluster and as the leader of that cluster. At fixed intervals, all nodes broadcast a vector, (node ID, cluster ID, path to the leader, number of nodes in cluster), to their one-hop neighbors. Each node also sends a hello message to the leader node of its cluster. By counting the hello messages, the leader determines the current size of the cluster. A node updates this size from a neighbor with the shortest path to the leader. If no such neighbor exists, the field is set to infinity.

Nodes dynamically update their cluster ID based on their neighbors' broadcast. After waiting for a random period of time, each node decides whether to update its cluster ID with probability P_{switch} . P_{switch} is lower when the number of nodes in a cluster is near n . The updated cluster ID of a node is a random variable that takes on values from the set of cluster IDs of neighboring nodes and a new cluster ID. The value is chosen in a way that favors the more popular IDs and better cluster sizes.

The prospective clustering may vary frequently and take on suboptimal clustering configurations. However this behavior is necessary to discover better clustering configurations. The actual clustering is updated from the prospective one at a slower rate. The more stable actual clustering is used for channel allocation and routing. The algorithm also chooses gateway nodes so exactly one link exists between any two neighboring actual clusters.

C. Channel Allocation

One may formulate channel allocation as an integer program. However, this approach is unsuitable for distributed implementations because its solution requires global network knowledge.

We propose to use a simulated annealing approach to solve our combinatorial optimization problem as in [7] [13].

By an intra-cluster link state protocol, each node can determine the set of reachable nodes within its cluster and the routing distances. Whenever a node becomes idle, it changes the channel on one of its interfaces and tests how good the new channel is. This goodness, called potential in simulated annealing literature, is defined by the number of neighbors and how well it is connected to the rest of the network. If its potential is reduced, it adopts this new channel; otherwise, it reverts to the original one with a probability depending on the magnitude of the increase. As the adaptation progresses, this probability is gradually discounted by a parameter called temperature to achieve convergence.

A node stays on a channel for a random period before switching. The length of this period depends on the potential and the temperature. The higher they are, the faster a node tends to switch. This design stabilizes the distributed algorithm and speeds convergence. When the network topology changes suddenly, a surge in the potential of affected nodes triggers the temperature to be *reheated* and the adaptation process restarted.

D. Routing

The presence of different priority levels in many disaster relief and military applications prompts us to use different routing algorithms for flows of different levels:

1. Urgent Messages (UM) are sent over shortest paths in terms of end-to-end transmission delay. Currently, a static link cost of $1/C$ is used, where C is the link speed, but we plan to incorporate queueing delays too.
2. Urgent Flows (UF) use shortest paths with link cost given by $1/C + \lambda I$, where C is the link speed, I the amount of bandwidth used by interfering links, and λ a tuning parameter. I is calculated by information obtained through a link state protocol (see below) and is the rate at which a link is busy either sending or receiving data. Note that its value is not obtained by measurements, but by information contained in flow setup and tear-down requests. The parameter λ captures the tradeoff between resource availability and delay optimality.
3. Regular Flows (RF) uses the least loaded routes. Because interference constraints are coupled, the resulting program is NP-hard [10] and hence a *myopic* algorithm is unlikely to work. For these reasons, upon arrival of a new RF call, our algorithm solves a linear program that maximizes available capacity to find the route for the flow. solved. The solution of a linear program splits a single flow over multiple paths in general. We discretize the solution by selecting the least loaded link on each hop, starting from the source node¹.
4. Best Effort (BE) use shortest hop count paths (unit link costs), as is the case for signaling traffic.

Using shortest path algorithms for UMs and UFs avoids the high route computation delay associated with RFs. The use of an additive interference link metric for UF is justified because these flows account for only a small fraction of the traffic. On the other hand, this is certainly not the case for RFs.

UF packets have priority over RF and BE packets, while UM packets have priority over UF. When the interference metric I

¹Loops are prevented by appropriately perturbing the objective function

for a link is calculated, as is needed in the routing algorithm for UF, the bandwidth usage of RF and BE for *that link* is not taken into account. On the other hand, since we not assume any MAC layer QoS capabilities, I needs to include RF and BE bandwidth usage of interfering links.

When a node attempts to allocate link bandwidth for a UF or an RF, it first requests approval from all of its interfering neighbors. The request is granted by a neighbor, if and only if it does not violate the interference constraints on any of that neighbor's links. If the allocation request is for UF or RF traffic, then a neighbor might need to inform its neighboring nodes to throttle back their BE traffic by an appropriate amount.

For scalability reasons, we define a two level routing hierarchy. At the lower level, node clusters exchange network state information within the same cluster. This is done by using a link state protocol similar to OSPF but with additional information such as per class bandwidth usage of interfering nodes. Cluster gateways export a topology summary of the cluster to other gateways in different clusters using another link-state protocol.

Link failures due to mobility are handled at two levels. The gateways of the cluster containing the failed link try to reroute any transit flows crossing the cluster if possible. Flows that cannot be rerouted locally are handled by the gateways of the source and destination nodes. At both levels, many flows may need to be rerouted by the same node. In these cases, the new paths are computed solving a single optimization problem for each class: Urgent Flows are routed first, then Regular Flows.

III. SIMULATION RESULTS

A. Simulation Framework

To evaluate the performance of our proposed system, we implement a simple time-driven flow level simulator in MATLAB [4]. The simulator models the arrival and departure of flows with different requirements, and the volatile link connectivity among the nodes due to mobility. However, individual packets are not simulated. All control information is kept in a global data structure in the simulator. To simulate the distributed nature of the system, different logical instances of various algorithms can only access part of the data structure.

In our mobility model, the nodes belong to static administrative groups. The members of each group roughly travel together, but the relative positions of the nodes vary over time randomly. The center of gravity of each group follows a random walk over a square area. The speed and the direction of travel of the nodes can change by a small amount in every time step. As the groups move, they may overlap. The simulator models the wireless links of various speeds. Each link can take on a speed from a set of predefined speeds based on the distance between the two interfaces.

B. Clustering Algorithm

We compare the results of our distributed clustering algorithm with an optimal solution computed by a slow, centralized, exhaustive search. We generate scenarios of 25 nodes with an ideal cluster size n of 5 with various average node degrees. The target cluster size used in this section is also 5. The network is stationary. We run our algorithm for 20 iterations.

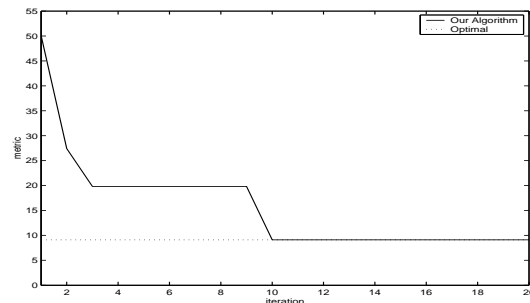


Fig. 1. Metric after each iteration.

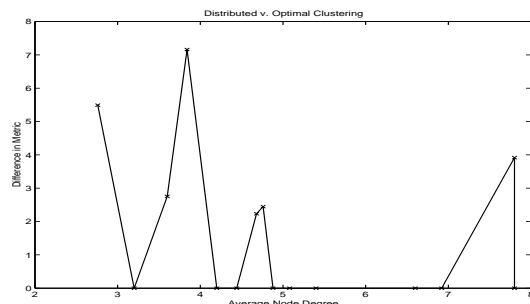


Fig. 2. Performance difference between ours and the optimal scheme for scenarios of different average node degrees.

Define the mean square error of a clustering (MSE) to be the sum of the squared difference between each cluster size and n , divided by the total number of clusters. We define the goodness metric (or simply the metric) of a particular clustering to be the sum of MSE and the total number of intercluster links. Fig.1 shows the how this metric changes after each iteration for a single scenario. The optimal metric is shown by the dotted line. Our algorithm does not always converge to the optimal. Fig.2 shows the performance difference for various scenarios denoted by average degree. For these scenarios, the difference is small even after a limited number of iterations especially when the optimal cluster size is near the target cluster size. However, our distributed algorithm computes the clustering in seconds while the centralized exhaustive search take days.

C. Clustering and Channel Allocation

To study the interaction between clustering and channel allocation algorithms, we simulate a 100-node network divided into 10 groups. Nodes move in groups as described earlier and reorganize themselves using the clustering algorithm. Once new clusters are formed, nodes execute the channel allocation algorithm to find the best way to connect with each other. Fig.3 shows the average potential of all nodes over time. In this scenario, nodes stop moving after 70 time steps (each step corresponds to 2.5 seconds). The channel allocation algorithm is able to at least maintain the performance of the network when nodes are moving. After that, it gradually “cools” down and settles into its steady-state, as indicated by the downward trend in the last part of the curve. For the same scenario we also show the gain in the average throughput per node that our algorithm can achieve in Fig.4. The curve at the bottom of the plot shows the average throughput per node when all interfaces are assigned to

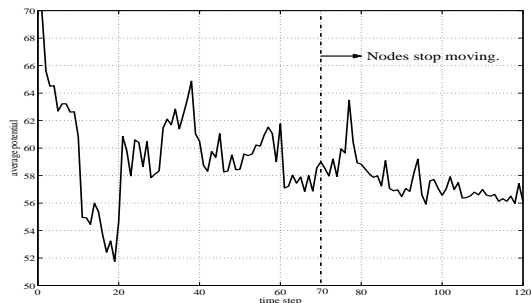


Fig. 3. Average potential vs. time

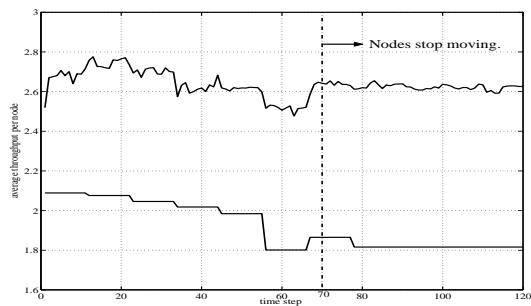


Fig. 4. throughput vs. time

the same channel. So in this experiment our algorithm achieves a factor of 1.44 increase in average throughput per node.

D. Routing

In this section, we present the simulation results of our routing algorithm running on top of a stationary network. We compare our routing algorithms with one that does not account for the coupling between interference among neighboring links. Namely, each time a Regular Flow arrives it is routed over the path with the highest available bandwidth without considering interference between neighboring links. This algorithm can be seen as a myopic version of our RF algorithm, in that the available bandwidth of each link is independent of other links. A similar myopic algorithm has been used in [3].

The simulated network consists of a single cluster of 20 nodes. Each node has a random number of radios that varies from one to three, and each radio can use a fixed frequency out

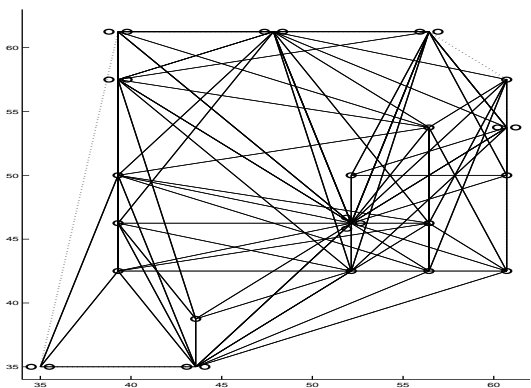


Fig. 5. Network topology used in simulation.

of three possible. The topology was randomly generated and is depicted in Fig.5. Regular flows arrive randomly in the network at a very high rate averaging 1 flow each second.

We ran two simulations: one using our algorithm and another one using the myopic one, with the same arrival pattern (generator seed) for 500 seconds of simulation time. Our algorithm accepted rejected 122 flows out of 613 (rejection rate 20%), while the myopic algorithm rejected 142 (rejection rate 23%).

IV. CONCLUSION

In this paper, we have presented a QoS routing system that considers both channel allocation and clustering. Our simulation results also shows fast convergence of our clustering algorithm producing good sized clusters after about 20 iterations. Additionally, once the clusters reach the target cluster size, the clustering remains stable when network topology is stable. In networks with mobility (when the neighbors of each node vary slightly per iteration), the algorithm is able to adapt to changes and assign clusters accordingly.

Finally, we have presented a set of fast routing algorithms that provide a diverse set of QoS guarantees to different applications. This was made possible by introducing a two level routing hierarchy that decomposes the routing computation into several lower complexity subproblems. This hierarchical approach is necessary in a large wireless network where the provision for QoS in the presence of interference constraints significantly increases the complexity of the problem.

This work is part of a joint project between Cisco and UC Berkeley. David Jaffe is the lead contributor from Cisco. The UC Berkeley team includes Venkat Anantharam, Eric Chi, Bill Hodge, Zhanfeng Jia, David Tse, and Pravin Varaiya. We would especially like to thank DARPA (contract no. N66001-00-C-8062) for their funding support for the project.

REFERENCES

- [1] S. Basagni, I. Chlamtac, and A. Farago. A Generalized Clustering Algorithm for Peer-to-peer Networks, July 1997.
- [2] S. Chen and K. Nahrstedt. An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions. *IEEE Network Magazine*, 1998.
- [3] S. Chen and K. Nahrstedt. Distributed quality of service routing in ad-hoc networks. *IEEE JSAC*, 17(8), 1999.
- [4] The MathWorks Inc. Matlab 6.5 release 13, June 2002.
- [5] I. Katzela and M. Naghshineh. Channel assignment schemes for cellular mobile telecommunications: A comprehensive survey. *IEEE Personal Communications*, pages 10–31, 1996.
- [6] R. Krishnan, R. Ramanathan, , and M. Steenstrup. Optimization Algorithms for Large Self-Structuring Networks. In *Proc. IEEE INFOCOM, 1999*, New York, March 1999.
- [7] P. Laarhoven. *Simulated Annealing: theory and applications*. Kluwer Academic Publishers, 1987.
- [8] C. R. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks, September 1997.
- [9] C. R. Lin and J-S. Liu. Qos routing in ad hoc wireless networks. *IEEE JSAC*, 17(8), 1999.
- [10] E. Magana, D. Morato, H. W. So, B. Hodge, J. Walrand, and P. Varaiya. A Wireless Overlay Network with QoS Capabilities, December 2002.
- [11] A. McDonald and T. Znati. A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks, August 1999.
- [12] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance vector routing (dsv) for mobile computers. In *ACM SIGCOMM '94*, London, UK, September 1994.
- [13] B. Preneel and J. Walrand. Convergence of a Quasi-Static Frequency Allocation Algorithm. In *technical report to ERL at U.C. Berkeley*, 1995.
- [14] P. J. M. van Laarhoven and E. H. L. Aarts. *Simulated Annealing : Theory and Applications*. Kluwer Academic Publishers, 1987.