Adaptive Routing For Ad Hoc Wireless Networks Providing QoS Guarantees

Gautam Barua and Indraneel Chakraborty
CSE Dept., Indian Institute of Technology
North Guwahati, Guwahati 781031, India.
e-mail: `gb@iitg.ernet.in`

ABSTRACT

A cluster-based route discovery and dynamic route management protocol for ad hoc networks with inaccurate information for given QoS (Quality of Service) requirements has been proposed and compared with existing routing protocols. The proposed scheme is much more scalable, the route discovery is faster (O($logN$) against O(N), N being the number of nodes) and it also guarantees QoS - a feature which has not been incorporated in any existing protocol. Also, features such as local dynamic route maintenance, loop avoidance load balancing, delay bounded routing and guaranteed rate routing have been included in the architecture of ACRQ. ACRQ can also work in case of inaccurate information by assigning weights according to the probability of each information being true. This information is then used for routing decisions. Furthermore, ACRQ takes advantage of the MAC characteristics of present wireless technologies such as master-driven communication in Bluetooth. Master-driven systems encourage cluster based routing, which is also helpful for scalability. A distributed algorithm for clustering and election of cluster heads (in ACRQ protocol) with only one round of message exchange has been proposed as well.

I. INTRODUCTION

With the present increase in the popularity of mobile networks, it is imperative that ad hoc networks support quality of service for real-time traffic. In this project, routing protocols designed for ad hoc networks have been studied and their characteristics have been compared. It was found that the present protocols do not consider network parameters other than the shortest path for routing decisions.

Hence a new protocol for faster route discovery and route maintenance has been proposed: Adaptive Cluster-based Routing with QoS support(ACRQ). Each characteristic of the protocol has been compared with the existing protocols to show superiority of the former. Features such as load balancing, loop avoidance, delay bounded and rate assured routing and dynamic maintenance of routes have been incorporated in the proposed scheme. We introduce a distributed clustering algorithm which has been implemented as a part of ACRQ. This algorithm can also be used at its own merit for other purposes such as scatternet creation in ad hoc networks. ACRQ performs even better when run on prevalent MAC protocols which are master-driven.

II. MOTIVATION FOR OUR WORK

Some of the recent works [5], [16] point to the need of a QoS based scheme. However, none of the current important routing protocols [4], [6], [7], [8], [12], [13], [17], [18], [19], [20], [22] take into consideration that ad hoc networks may have inaccurate information due to very frequent changes in the network. Most of them also do not give QoS parameters their due importance in routing. Furthermore, except for CGSR and ZRP, any other protocol can not take advantage of the MAC characteristics of present technologies in this field such as master-driven communication in Bluetooth [1]. Master-driven systems encourage cluster-based routing which is helpful for scalability.

Full version of this paper available at hercules.lcs.mit.edu/acrq.pdf.

III. A NEW PROTOCOL: ACRQ

The proposed algorithm, ACRQ, considers QoS parameters of rate and delay of a connection while deciding routes for a new connection and while improving the routes of already established connections. ACRQ is a distributed cluster-tree based algorithm to determine the route of a new connection with a given minimum rate on each link of the path, while satisfying the end-to-end delay constraints.

A. Complexity of QoS Routing

We give below a quick formal proof of the non-triviality of the problem at hand.

Theorem 1: *The problem of determining the route of a new connection with a given minimum rate r on each link of the path while satisfying the end-to-end delay constraint 'd' is NP-complete.*

Proof: We are going to reduce the problem at hand, to a shortest weight-constrained path problem (SWCP), which is known to be NP-complete. Given a graph with two positive values, a_l (delay of link) and b_l (bandwidth of link), associated with each link l and a positive bound B, the shortest weight-constrained path problem is to find a path that:

$$min \sum a_l \tag{1}$$

$$\sum b_l \leq B \tag{2}$$

To transform SWCP into the problem at hand, the delay on each link can be directly mapped to b_l. The delay constraint d is the same as B. Because, delay d_l and rate r_l on each link are inversely related, then a path that maximizes the probability of not exceeding the delay constraint by increasing the minimum rate r guaranteed over the link, is also a path that minimizes $\sum a_l$ (which corresponds to r_l) while obeying $\sum b_l \leq B$ (which corresponds to d_l). Thus the problem at hand is also NP-complete. Hence solutions such as ACRQ which try to attain near optimality at bounded costs of route discovery are justified. The proposed algorithm therefore reduces the state space of the route, by clustering nodes in the network.

As we are assuming inaccurate information in the network, hence the metric to decide the path of the new connection will be the *probability* $p(r)$ of availability of a given rate r on the link l. The higher the value of $p_l(r)$, the higher the weight of the link l. Hence the metric of route discovery is:

$$w_l(r) = -\log p_l(r) \tag{3}$$

B. Overview of the ACRQ

The working of ACRQ can be formalized in a step by step manner as follows:

- The whole ad hoc network divides itself into clusters. The formation of clusters is achieved by a distributed algorithm which is proposed in section IV. These clusters form a hierarchy by grouping among themselves to form super-clusters. Thus a tree is established. This is required for scalability in case of thousands of nodes.
- The master of each cluster collects link information from each node of the cluster to find the maximum delay d_{max} and minimum rate r_{min} which the nodes can provide to any connection passing through the cluster. The values of d_{max} and r_{min} provided are qualified by the logarithm of the probability that the resources are still available ($w(r)$). These values are available at the clusterhead. Information about clusterheads is available at a super-clusterhead and so on.
- Each node contains only one-hop link information required to forward packets on the route to its clusterhead. This eliminates the overhead of source routing from data packets. Packet forwarding information required for different connections is collected on-demand. This too is only a single hop information showing the next node in the chain. Source routing becomes unnecessary because the clusterhead has information about each node in its cluster. Hence, route maintenance, loop avoidance and load balancing can be handled by the clusterhead. Also routing loops as experienced in TORA are avoided because of a central control. In case of route maintenance involving more than one cluster, the super-clusterhead is involved in route maintenance.
- Whenever node S situated in a cluster requests a link to any node, the ROUTE REQUEST is directed to the master M_S of the cluster concerned, who directs it to only the master of the super-cluster. The information travels through the tree and reaches the master M_D of the cluster containing the destination node or it reaches to one of the master nodes M_T which belongs to the set of masters S_M at the highest level. In the former case, the master M_D sends a ROUTE REPLY that contains the number of hops necessary to reach D and the sequence number for D most recently seen by the node generating the REPLY. Otherwise the master M_T performs a route search by multicast among the set of masters at this top level S_M of the hierarchy to find a route to a clusterhead containing the destination node. Each master that participates in forwarding this REPLY back toward the originator of the ROUTE REQUEST (node S), creates a *forward value* to D. The state created in each master node along the path from S to D is a hop-by-hop state; that is, each node remembers only the hop to the next master and not the entire route, as would be done in source routing.
- Each master on the route sends only the best path to the previous master if it gets a ROUTE REPLY from more than one neighboring masters. The choice between alternative routes is made on the basis of the value of $w(r)$ (in case more than one route satisfy the d_{max} and r_{min} criteria) for the requested QoS parameters.
- When the route is established through super-clusters, it is gradually fine-tuned by choosing nodes (locally within the super-cluster) which have higher $w_l(r)$ available, than the ones which are presently on the route. This can be easily done locally without the information of the source S or the destination D. For example, suppose there is a route from S to D via nodes A, B and C in that order, while node E has a better value of $w_l(r)$ than B for the connection. Then B sends a message to its neighbors A and C to update their tables to send data to E for that connection in place of sending it to B. This ensures dynamic load balancing and lesser network overheads during network route discovery. This internal re-ordering of nodes can be managed by the master or a node especially in charge of internal load balancing (who may or may not be the master).
- Network Instability (the condition when sub-optimal paths are taken even if optimal paths are available) can be locally avoided if masters check their own clusters for alternate paths and avoids sub-optimal paths if unnecessary (i.e., in case of no congestion).

As stated in the aforementioned algorithm, hierarchical clustering is employed for ACRQ. Route discovery follows this hierarchy and contacts the master of the super-cluster in case the routing information is not available at the immediate master, and further up the chain until the information is found, or the ROUTE REQUEST reaches the root of the chain. Assuming that the branching at each level of the tree thus formed is uniform, we have a tree of depth O($logN$). In the worst case, the ROUTE REQUEST will have to travel O($logN$) hops and the ROUTE REPLY will also travel as many hops. This gives a communication complexity of O($2logN$) for ACRQ.

IV. CLUSTERING

For efficient communication between nodes, ad hoc networks are usually organized into clusters [3], [8], [9], [11], [15] where each cluster has a clusterhead. Clustering is a key issue in our routing protocol ACRQ as well. Clustering based protocols have an advantage over the other kinds as emerging technologies such as Bluetooth [1] are completely clustered and are master-driven communication systems. In Bluetooth, which is for indoor wireless picocellular environments, each cluster is a star (*piconet*), with the master at the center of the star. The master node controls the traffic to all the slaves on the wireless channel. A connected set of piconets is called a *scatternet*. The formation of clusters, and the related leader election problem have been investigated in several papers [2], [21], [24].

However, these solutions cannot be used to solve the problem we address, where we require the communication between nodes in different clusters to be through 'bridge' nodes, which are non-clusterhead nodes common to at least two clusters [1]. Moreover, for the distributed case, we do not assume full knowledge of the topology, and allow only one round of message exchange between neighbors, which results in the nodes having 2-hop neighborhood information.

We model the set of nodes as a graph, with an edge between a pair of nodes if they are in radio range of each other. It is assumed that a device can discover other devices within its radio-range using device discovery protocols. The aim of the scatternet formation problem is to get a minimum set of connected star-shaped clusters of bounded size. The connection between the stars is to be made through non-center nodes, or bridges, which, along with the clusterheads form the backbone of the network.

However, if there is no such connected scatternet, then we allow direct communication between two center nodes or masters. This is because, in Bluetooth, the master device controls the traffic to all the slaves, and a node can be active in only one piconet at a time. This means if a device (which is a master in piconet A and slave in piconet B) is active as a slave in piconet B, then during this time, the entire piconet A, for which this device is the master, has to be idle. Similarly, a device which is a slave in more than one piconet is active in only one of the piconets at a time, and hence the number of piconets to which a slave can be common should be limited. We develop a distributed heuristic algorithmic which assumes that each node knows its 2-hop neighborhood.

A. Distributed Heuristic Algorithm

For a meaningful application of any algorithm for ad hoc network formation, the algorithm has to be completely distributed since usually, no central infrastructure exists in such networks. In order to minimize the number of message exchanges, while providing some useful information to the nodes about their neighborhood, we let each node exchange one round of messages containing immediate neighborhood information, with all its neighbors.

Once the 2-hop neighborhood information is known, the following distributed algorithm is executed at every node u. The algorithm proceeds in two phases: the cluster formation phase, and the cleanup phase. In the cluster formation phase, the nodes get marked as master/slave/bridge and star clusters are formed with bridge nodes common to several masters. In the cleanup phase, each bridge node exchanges its master information with all its masters and any redundant links are removed, while ensuring that the set of masters known to it is connected. This is to avoid a node being common to more clusters than necessary. This will help in having different paths for different connections decreasing the probability of any bridge becoming a bottleneck for connections.

The cluster formation algorithm is outlined in the following steps:

1. If $degree(u)$ is higher than the degrees of all neighbors of node u, then u becomes a master and picks k lowest degree neighbors as slaves. Each node also checks if the effective degrees of any of the remaining nodes is zero. If so, then the node concerned becomes a master to ensure connectivity.
2. Else

The following steps are repeated for every neighbor v which has a higher degree than u.

(a) If $degree(v)$ is the highest in the neighborhood of v

i. If u is among the k smallest degree neighbors of v, then u becomes a slave of v. Each lower degree neighbor of u, (let that be called w), which is of greatest degree in its own neighborhood (without considering u), is made a master.

ii. Else u repeats the algorithm for the remaining (unmarked) neighbors.

(b) Else

i. If the $degree(u)$ is the highest in the neighborhood of u (excluding the edge from v to u), then u becomes a master.

ii. If $degree(v)$ is the highest in the neighborhood of u, then u becomes a slave of v.

Special case in the above algorithm are:

1. In steps 2(a) - 2(b), a node becomes a slave only if it does not become a master by any other rule.
2. A node which receives a "become master" message always becomes a master even if it had chosen to become a slave by degree concerns. This attempts to avoid network partitions, even though the number of masters might increase.
3. When two devices have equal degree, the tie is broken using the unique IDs of the nodes.

The cleanup phase works as follows:

1. At the end of this step, each bridge node sends a list of all its masters to each of its neighbors.
2. A bridge node removes extra links to each of those masters who are already connected to any of its other masters, by a link not involving the bridge. This step helps the bridge node to avoid being a bottleneck in the network.
3. If there are two masters which are adjacent to each other and there is no common slave, then a master-master edge is added to the scatternet

Clearly, the above distributed algorithm ensures that every node is marked as either a master, slave or a bridge. Special case 2 might cause the number of masters to increase in some cases while attempting to avoid network partitions. Connectivity of the entire scatternet cannot be guaranteed, as a bounded degree graph formation is not even feasible for some graphs (eg. a tree with branches$>$ than the bound). Special case 3 ensures that there are no deadlocks.

V. PERFORMANCE OF ACRQ AGAINST BENCHMARK PROTOCOLS

A. Evaluation Methodology

The overall goal of our experiments was to measure the ability of the routing protocols (including ACRQ) to react to network topology change while continuing to successfully deliver data packets to their destinations. Our protocol evaluations are based on the simulation of 50 wireless nodes forming an ad hoc network, moving about over a square ($670\text{m}\times670\text{m}$) flat space for 600 seconds of simulated time. For the results shown here, the physical radio characteristics of each mobile node's network interface, such as the antenna gain, transmission power, and receiver sensitivity, were chosen to approximate the Lucent WaveLAN [23] direct sequence spectrum radio. However, experiments have also been done with the Bluetooth [1] physical characteristics and MAC layer.

A.1 Movement Model

Nodes in the simulation move according to a model that we call the "random waypoint" model [14]. Each node begins the simulation by remaining stationary for *pause time* seconds. It then selects a random destination in the 670 m $\times670$ m space and moves to that destination at a speed distributed uniformly between 0 and a maximum speed of 20m/s. Upon reaching the destination, the node pauses again for *pause time* seconds, selects another destination, and proceeds there as previously described, repeating this behavior for the duration of the simulation. Each simulation ran for 600 seconds of simulated time.

A.2 Communication Model

All communication patterns were peer-to-peer, and connections were started at times uniformly distributed between 0 and 180 seconds. We did not use TCP sources because TCP offers a conforming load to the network, which means that it changes the times at which it sends packets based on its perception of the network's ability to carry packets. However, for measuring the performance at route discovery, the type of source (CBR, TCP etc.) is not important. The latter affects QoS parameters such as packet loss ratio. The protocols are being compared on Network Simulator *ns*. *ns* is a discrete event simulator developed by the University of California at Berkeley and the VINT project [10].

B. *Comparison Summary*

In this project, ACRQ has been compared with AODV, DSR, DSDV and TORA. In the simulations, we have chosen to remove CGSR from comparisons, as it basically shows similar characteristics as DSDV. This is because CGSR applies DSDV at cluster-level, thus causing the same problems as DSDV, but for a larger number of nodes. Figures 1 and 2 demonstrate the relative performance of the five protocols on our traffic loads of 20 sources.

All of the protocols drop lesser percentage of data packets when there is little mobility, converging to 0% loss when there is no node motion. From the available protocols, DSR and AODV perform particularly well, dropping less than 2% of the packets. In these scenarios, DSDV starts converging for *pause times* greater than 200 seconds. ACRQ maintains the lowest dropping ratio among the protocols.

C. *Packet Dropping Ratio Details*

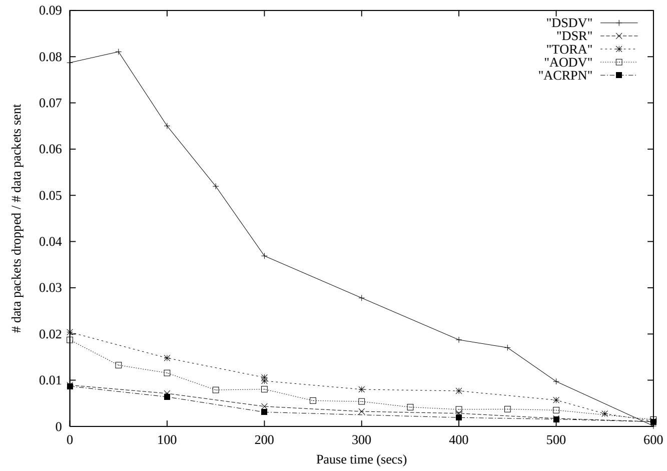

Fig. 1. Comparison of the fraction of application data packets dropped as a function of *pause time*

Figure 1 shows the fraction of the originated application data packets each protocol dropped, as a function of node mobility (*pause time*). DSR and ACRQ drop less than 1% packets. AODV drop 2% of packets at high mobility. DSDV fails to converge below pause time of 200sec, where it drops about 8% of its packets. This happens because a stale routing table entry directed them to be forwarded over a broken link. TORA performance is the worst among the on-demand protocols with more than 2% loss at high mobility. The majority of the packet drops are due to the creation of short-lived routing loops that are a natural part of its link-reversal protocol.

D. *Routing Overhead Details*

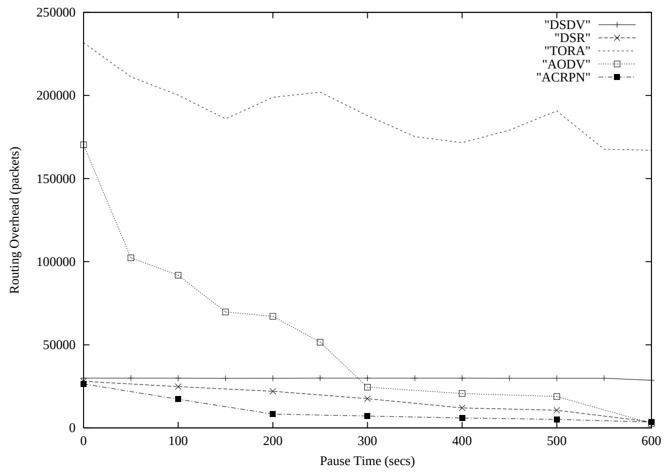

Fig. 2. Routing overhead in packets as a function of *pause time*

The overhead of ACRQ also includes clustering messages, one to disconnect from the old master and one to connect to the new master, for every change in topology. This is clear from the clustering algorithm of Section IV. The results shown include *clustering overhead* with the *routing overhead* for ACRQ. Figure 2 shows the number of routing packets sent by each protocol in obtaining the delivery ratios shown in Figure 1. DSR, AODV and ACRQ which use *only* on-demand packets and similar basic-mechanism, have comparable curves. However, the absolute overheads required by them are very different. This dramatic increase in AODV's overhead occurs because each of its route discoveries typically propagate to every node in the ad hoc network.

As ACRQ is a cluster-based routing protocol, route discovery information is propagated to the clusterheads only. The clusterheads are involved in routing decisions (not individual nodes). The decision about the propagation of route discovery information is dependent upon the hierarchy of clustering. Clusterheads in a particular sub-tree only will get the route-discovery information, thus further limiting routing overhead. This, however, might lead to sub-optimal routing if the clustering is not proper. Hence ACRQ depends upon the clustering algorithm as well. In these simulations, the clustering for ACRQ is only a level deep. Even better performance can be achieved by making the hierarchy deeper.

VI. CONCLUSION

Most of the present protocols do not consider network parameters other than shortest path, for routing decisions. This is insufficient for the present standards of quality of service expected from communication networks. Also, local dynamic route maintenance, loop avoidance, load balancing, delay bounded routing and guaranteed rate routing have been generally neglected in ad hoc routing schemes.

Hence, a new protocol (ACRQ) for route discovery and route maintenance has been proposed. All the features mentioned above as lacking in other protocols have been included in ACRQ. ACRQ, with its hierarchically clustered structure, has a message complexity of O($logN$), N being number of nodes. The minimum message complexity known to the authors in other protocols in O(N). ACRQ can also work in case of inaccurate information by giving weights according to the probability of each information being true. This information is then used for routing decisions. This is not possible in other protocols. The results in this work are from simulations done on *ns*.

Thus ACRQ should provide a complete solution to the requirements of the wireless networking industry which is now preparing itself to service the potentially huge market of real-time data traffic.

REFERENCES

[1] The Official Bluetooth SIG Website, 2001, http://www.bluetooth.com/

[2] H. H. A. Amara and J. Lokre, “Election in Asynchronous Complete Networks with Intermittent Link Failures”, *IEEE Trans. on Computers*, vol. 43, no. 7, pp.778–788, July. 1994.

[3] A. D. Amis, R. Prakash, T. H. P. Vuong and D. T. Huynh, “Max-min D-cluster Formation in Wireless Ad Hoc Networks”, *Proc. 2000 IEEE Infocom*, pp. 32–41.

[4] J. Broch, D. B. Johnson and D. A. Maltz, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks”, Internet-Draft, draft-ietf-manet-dsr-00.txt. March 1998.

[5] S. Chakrabarti and A. Mishra, “QoS issues in ad hoc wireless networks”, *IEEE Communications Magazine*, vol. 39, no. 2, pp. 142–148, Feb. 2001,

[6] C. C. Chiang, H. K. Wu, W. Liu and M. Gerla, “Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel”, *Proc. 1997 IEEE Singapore International Conf.* pp.197–211.

[7] M. S. Corson and A. Ephremides, “A Distributed Routing Algorithm for Mobile Wireless Networks” *ACM/Baltzer Wireless Networks Journal*, vol. 1, no. 1, pp. 61–81, Feb. 1995.

[8] B. Das, E. Sivkumar and V. Bhargavan, “Routing in Ad Hoc Networks using a Spine,” *Proc. 1997 IEEE International Conf. on Computers, Commun., and Networks*, pp.1–20.

[9] R. Dechter and L. Kleinrock, “Broadcast Communication and Distributed Algorithms”, *IEEE Trans. on Computers*, vol. C, no. 35, pp.210-219, 1986.

[10] K. Fall and K. Varadhan. “ns Notes and Documentation” *The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC* November 1997. Available from http://www-mash.cs.berkeley.edu/ns/ or http://www.isi.edu/nsnam.

[11] M. Gerla and J. T. C. Tsai, “Multicluster, Mobile, Multimedia Radio Network,” *ACM Baltzer Journal of Wireless Networks*, vol. 1, no. 3, pp.255–265, Oct. 1995.

[12] Z. J. Haas and M. R. Pearlman, “The Zone Routing Protocol (ZRP) for Ad Hoc Networks,” Internet Draft, draft-ietf-manet-zone-zrp-02.txt, June 1999

[13] T. C. Hou and T. J. Tsai, “An access-based clustering protocol for multihop wireless ad hoc networks”, *IEEE Journal on Selected Areas in Communications* vol. 19, no. 7, pp.1201–1210, July 2001.

[14] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks”, *Mobile Computing*, sec. 5, pp. 153–181. Kluwer Academic Publishers, 1996.

[15] R. Krishnan, R. Ramanathan and M. Steenstrup, “Optimization Algorithms for Large Self-structuring Networks”, *Proc. 1999 IEEE Infocom*, pp. 71–78.

[16] C. R. Lin and J. S. Liu, “QoS Routing in Ad Hoc Wireless Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1426–1438, August. 1999.

[17] V. D. Park and M. S. Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks”, *Proc. 1997 IEEE Infocom*, pp. 1405–1413.

[18] C. E. Perkins and P. Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector routing (DSDV) for Mobile Computers”, *Proc. 1994 ACM Sigcomm Conf. on Commun. Architectures, Protocols and Applications*, pp. 234–244.

[19] C. Perkins, “Ad Hoc On Demand Distance Vector (AODV) Routing”, Internet-Draft, draft-ietf-manet-aodv-00.txt, Nov 1997.

[20] R. Ramanathan and M. Streenstrup, “Hierarchically Organization, Multihop Mobile Wireless Networks for Quality of Service Support”, *Mobile Network and Applications*, vol. 3, pp. 101–19, 1998.

[21] G. Singh, “Leader Election in Complete Networks”, *SIAM Journal of Computing*, vol. 26, no. 3, pp. 772–785, 1997.

[22] C .K. Toh, “Associativity-Based Routing for Ad hoc Mobile Networks” *International Journal Wireless Personal Communications*, Vol. 4, No. 2, pp. 1–36, 1997.

[23] B. Tuch, “Development of WaveLAN, an ISM Band Wireless LAN”, *AT&T Technical Journal*, vol. 72, no. 4, pp. 27–33, July-Aug. 1993.

[24] J. Wu and H. Li, “On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks”, *Proc. 1999 ACM International Workshop on Discrete algorithms and methods for mobile computing and communications*.