On Fault Tolerant Ad Hoc Network Design

Wassim El-Hajj
College of Information Technology
UAE University
United Arab Emirates
welhajj@uaeu.ac.ae

Hazem Hajj
Faculty of Engineering and Architecture
American University of Beirut
Lebanon
hh63@aub.edu.lb

Zouheir Trabelsi
College of Information Technology
UAE University
United Arab Emirates
trabelsi@uaeu.ac.ae

ABSTRACT

Minimal configuration and quick deployment of ad hoc networks make it suitable for numerous applications such as emergency situations, border monitoring, and military missions, etc. For such ad hoc networks to fulfill their mission in a timely manner, they should be able to establish a connection between nodes and to maintain this connection until the communication halts. Establishing a connection is achieved by using a routing protocol, and maintaining it is achieved by having a resilient fault tolerant network. In this paper, we propose a network design scheme that incorporates these features. We first propose a special network topology that is unique in terms of how nodes are interconnected. After constructing the initial topology, we propose a distributed routing protocol that allows any two sites to communicate by traversing at most 2 nodes regardless of the network size. We conducted both simulation study and theoretical analysis; the results show that the proposed scheme is resilient to network dynamics and has high quality as well as efficient routing.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication, Network topology*; C.4 [**Computer Systems Organization**]: Performance of Systems—*Fault tolerance*

General Terms

Design

Keywords

Network design and planning, Ad Hoc Networks, Fault Tolerance

1. INTRODUCTION

An ad hoc network is an infrastructure-less network with distributed routing and built in maintenance. Its quick and cost-effective deployment makes it suitable for many applications that are mission critical, such as emergency situations. In such scenarios continuous mode of operation and speed are very crucial. The need for such applications has stimulated strong interest in achieving better Quality of Services (QoS). In this paper, we work on enhancing the QoS by (1) suggesting a network design that is resilient to errors and by (2) introducing an efficient distributed routing protocol that is very fast and needs minimal storage space. So our work tackles a network design-planning problem and a routing problem.

Network design and planning is an extremely important task which must be performed before the establishment of the network. If this task is to be performed with QoS support, it becomes a multi-step process that involves the identification of the following aspects: (i) identification of network node location; (ii) definition of the link topology; (iii) definition of a routing strategy accounting for external input traffics; (iv) capacity allocation to the links so that suitable QoS metrics are fulfilled. In our work we address the first three steps.

For the first two steps, if the ad hoc network was not yet deployed, we choose the appropriate locations of the network nodes, then we interconnect them in such a way to satisfy our proposed network design. After that, we use our proposed routing protocol to route traffic between network nodes. In case the ad hoc network is already deployed i.e. the nodes' locations are already known, we keep it for future work to propose a way to map our suggested topology to the already existing one. The routing scheme remains the same.

For the third step (routing strategy), we propose an on-demand distributed routing protocol which improves the overall throughput of the network without adversely affecting the requested services. On-demand protocols are usually preferred in Ad Hoc networks because of their lower overhead. Several distributed routing approaches have been proposed in literature.

In [1], a distributed shortest-path routing algorithm based on the Ford- Fulkerson method was proposed. In the algorithm, each node maintains a routing table that lists the cost of the shortest path and the next node in the path to every other node in the network. Each node broadcasts the cost information to its neighbors either periodically or whenever path costs change. The shortest path to a given destination is then computed based on the partial network topology information maintained inside of each node.

In [2], an on-demand protocol named Ad hoc On-demand Distance Vector Routing (AODV) was proposed. AODV

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IWCMC'09 June 21–24, 2009, Leipzig, Germany
Copyright © 2009 ACM 978-1-60558-569-7/09/06 ...$5.00.

protocol has the advantage of efficiently using the bandwidth by minimizing the network load for control and data traffic and ensure a loop free routing. However, this protocol presents the drawback of using routing caches to reply to route queries, which results in an uncontrolled replies and repetitive updates that are flooded all over the network.

Another well-known source based on-demand routing protocol is Dynamic Source Routing (DSR) [3, 4]. Nodes within the network are required to maintain route caches that contain all information about the already established routes. Entries in the route cache are continuously updated as new routes are discovered. The protocol consists of two major steps: route discovery phase and route maintenance phase. The DSR protocol has the drawback of using routing caches to reply to route queries, which results in an uncontrolled replies and repetitive updates that are flooded all over the network. Furthermore, DSR, also suffers from the scalability problem.

Temporally-Ordered Routing Algorithm (TORA) [5] is a highly adaptive, loop-free, distributed routing algorithm based on the link reversal concept. TORA was designed to operate in a highly dynamic mobile networking environment. It is source initiated and provides multiple routes for any desired (source, destination) pair. The major drawback of TORA is the count-to-infinity problem, which is caused by the use of internodal co-ordination.

A totally different approach in mobile routing is presented in [6, 7]. The Associativity-Based Routing (ABR) protocol is free from loops, deadlock, and packet duplicates, and defines a new routing metric for ad-hoc mobile networks. This metric is known as the degree of association stability. In ABR, a route is selected based on the degree of association stability of mobile nodes.

Our on-demand distributed routing protocol differs from all the protocols discussed above, since it is designed to operate on a special topology that satisfies several constraints. As a result of these constraints, the resultant network topology guarantees the existence of a path that is maximum one intermediate hop between any pair of nodes i.e. each node can communicate with any other node by traversing either 1 or 2 hops. Therefore the routing path between 2 nodes is actually the shortest path between them. Moreover, the suggested routing protocol achieves a better routing performance by considerably reducing the routing table size that needs to be maintained by each node.

The rest of this paper will be structured as follows. Sections 2 and 3 describe our network topology specifications. Section 4 discusses the on-demand distributed routing scheme. Section 5 presents our results and section 6 concludes the paper and discusses future work.

2. NETWORK DESIGN

A fully meshed network is the ultimate in fault tolerance and is very easy to manage and control. Routing in a fully meshed network is trivial since every node in the network is connected to every other node. Unfortunately, such a network is very costly to build. We, on the other hand, propose a network that achieves high fault tolerance with fewer number of links. Moreover, our design of the network topology guarantees that there exist a path of a maximum of one intermediate hop between any pair of nodes. We base our work on an idea used to obtain mutual exclusion of events in distributed environments [8].

In our proposed network design, each node in the network is assigned a communication set (C) satisfying the following constraints:

$$\forall n_i, \forall n_j : i \neq j, 1 \leq i, j \leq N :: C_i \bigcap C_j \neq \phi \qquad (1)$$

$$\forall i :\ 1 \leq i \leq N :: n_i \in C_i \qquad (2)$$

$$\forall i :\ 1 \leq i \leq N :: |C_i| = K \qquad (3)$$

Where N is the total number of nodes in the network and $K = \sqrt{N}$ in the optimal case. In addition to the above three constraints, a fourth constraint exists that states: any node n_j is contained in K number of communication sets C_i, $1 \leq i, j \leq N$. Once the above constraints are satisfied, the network topology is constructed by connecting each site with all nodes in its communication set. Figure 1 shows two examples of our network design.

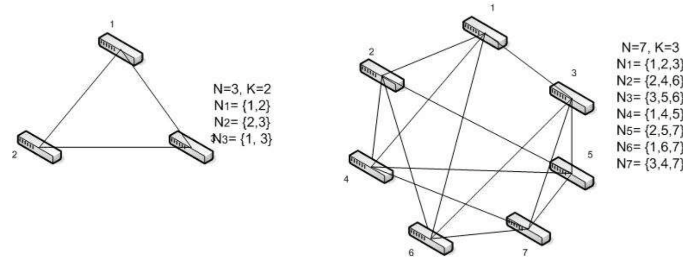

Figure 1: Example of two network topologies of size $N = 3$ and $N = 7$ nodes respectively

Equation 1 states there is at least one common node between the communication sets of any two nodes. For example, in figure 1, communication sets C_1 and C_4 have node n_1 in common. Also communication sets C_2 and C_7 have node n_4 in common. Equation 2 states that each node should belong to its own communication set i.e. node n_2 is part of communication set C_2. Equations 1 and 2 are necessary for the correctness of the topology.

Equation 3 suggests the size of each communication set to be equal to K. This condition implies that all sites have the capability of doing the same amount of work. This leads to load balancing which results in a more survivable network [9], [10], [11], [12], [13], [14]. The fourth constraint states that each site should be contained in K other sets, implying that all sites have equal responsibilities which is an important aspect in load balancing.

Therefore, given an ad hoc network of size N, our job is to generate a communication set for each node such that constraints 1 through 4 are satisfied. As a consequence, routing between nodes becomes very efficient since any two nodes will be either directly connected or a third node will exist to connect them. Next we discuss how these communication sets are generated.

3. COMMUNICATION SET CONSTRUCTION

We provide two different procedures to calculate the communication sets: (1) when $K + 1$ is the power of a prime number, and (2) when $K + 1$ is *not* the power of a prime number.

3.1 K+1 is power of a prime number

If $K+1$ is power of a prime number, the communication sets are generated in polynomial time using algorithm 1. Figure 2 illustrates the algorithm for a network with seven nodes ($N = 7, K = 3$). The topology represented by these sets is illustrated in figure 1. It is important to notice that the complexity of the algorithm is $O(N^2)$, which is explained with the fact that for each node, either the row, column, or diagonal is traversed. An extension of the same algorithm can be used to find the sets if $K+1$ is not a power of a prime. The same matrices will be filled with nodes indices, but some nodes will be duplicated in the matrix entries. This will be presented next.

Algorithm 1: Generating the communication sets when $K+1$ is power of a prime number

Data: N, K
Result: Communication Sets
begin
  Consider a matrix of size $(K-1) \times (K-1)$
  Generate K groups of $(K-1)$ nonintersecting sets
  begin
    $(K-1)$ nonintersecting rows
    $(K-1)$ nonintersecting columns
    $(K-2)$ of $(K-1)$ nonintersecting diagonals
    For different diagonals, jump 1 on each row (the main diagonal), jump 2, $\cdots$, jump $(K-1)-1$
  end
  Each number (out of the first K numbers) can be combined with each of the $(K-1)$ nonintersecting sets to produce $(K-1)$ of 1-element-intersected sets
end

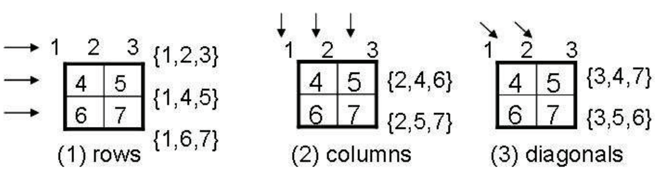

Figure 2: (1): node 1 is taken with the matrix rows. (2): node 2 is taken with the matrix columns. (3): node 3 is taken with the matrix diagonals.

3.2 K+1 is not power of a prime number

Unfortunately, the set assignment algorithm discussed in the previous section works for N number of sites where $N = K(K-1)+1$; K being the size of the communication sets. In this section, we present an algorithm for computing the Optimum Network Configuration with minimum number of links for any network size N.

For the general case i.e. any number of sites, the generation of the communication sets is tricky. When $N \neq K(K-1)+1$ we need to find the value of a number L such that $L = K(K-1)+1$ where $L > N$. So $L-N$ number of sites need to be duplicated in order to generate the matrix used for the generation of the communication sets (as in the previous case).

Algorithm 2 finds the number of duplications (dashes) needed for any given N. This algorithm takes as input the number of sites N, then outputs the number of dashes D and the communication set size K. For instance, if the network size is $N = 10$, then applying the above algorithm will give $K = 4$ and $L = 13$. Therefore $D = L - N = 3$ (figure 3).

Algorithm 2: Calculation of D and $|k|$

Data: N
Result: size of K, D
begin
  Calculate the size of communication sets $K = \lceil \sqrt{N} \rceil$ and let $L = K(K-1)+1$
  if $L < N$ **then**
    $K = K+1$
  $D = L - N$
end

A Brute Force algorithm duplicates all combinations of D dashes for N sites, then generates request sets for all the combinations, finally finds the minimum number of links for all the communication sets combinations. Considering this approach, the total number of combinations are N^D. However, the maximum number of dashes that we can have is $2(K-1)$. i.e. $D_{max} = 2(K-1)$, where $K = \sqrt{N}$. Therefore the complexity of the Brute Force algorithm is exponential and can be expressed as $N^{f(N)}$, where $f(N) = D$.

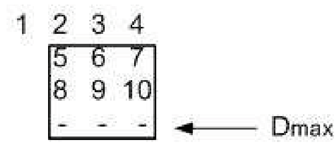

Figure 3: For N = 10, number of dashes = 3

Algorithm 3: Duplication Sites

Data: N, K
Result: combination of sites that should replace the dashes
begin
  for i = *1 to* D **do**
    for j = *1 to* N **do**
      substitute N for D_i
      compute communication sets for each nodes N_i
      calculate the number of links in the network
    find the value of N_i for which the number of links is minimum
    restrict N_i for D_i
end

The restriction based algorithm can further be improved in order to achieve less computation complexity. Rather than following the Brute Force technique, one dash is replaced at a time. In order to replace one dash, all values of N are considered. The value of N that minimizes the link cost is chosen to replace the dash. After replacing the dash, another dash is filled using the same procedure. This procedure continues until all dashes have been assigned.

Duplication sites for dashes are found using algorithm 3 that takes as input N and K then outputs the combination of sites that should replace the dashes.

Lines 4 and 8 can be performed in constant time $O(1)$; line 5 has complexity of $O(k * N)$; line 6, which consists of computing the number of links for N sites, takes $O(N)$ time; line 7, which consists of finding the minimum links, has complexity $O(N)$. When considering the for loops in lines 2 and 3, the overall complexity is $O(D * N * K * N)$. Substituting D by $D_{max} = 2(K-1)$, the complexity of the algorithm is $O(N^3)$ since $k = N$.

Considering the example above, where $N = 10$, the approximation algorithm works as shown in figure 4.

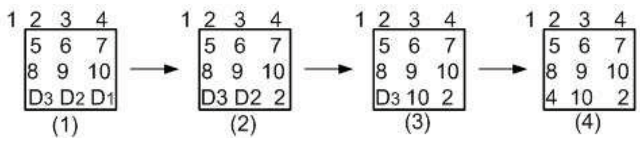

Figure 4: N=10. (1) 3 dashes need to be replaced. (2) D_1 is optimally chosen. (3) D_1 is kept, and D_2 is optimally chosen. (4) D_1 and D_2 are kept, and D_3 is optimally chosen.

Algorithms 1 through 3 are used to generate all communication sets without knowing which set belongs to which node. We propose algorithm 4 to assign each communication set to a corresponding node.

Algorithm 4: Assigning the Communication sets to the network nodes

```
  Data: Communication Sets
  Result: Node Assignment
1 begin
2   node 1 gets the first set
3   communication sets constructed from each row are
    assigned to the 2^nd node of the set.
4   node 2 gets the set of 2 and first column.
5   communication sets generated from each column are
    assigned to the 2^nd node of the set.
6   communication sets generated from each Jump-X
    diagonal are assigned to (X + 3)^rd node of the set.
7 end
```

The following steps occur when algorithm 4 is applied to a network of size $N = 13$.

- Line 2 assigns sets $\{1, 2, 3, 4\}$ to site 1
- Line 3 assigns set $\{1, 5, 6, 7\}$ to site 5, $\{1, 8, 9, 10\}$ to site 8, and $\{1, 11, 12, 13\}$ to site 11.
- Line 4 assigns set $\{2, 5, 8, 9\}$ to site 2
- Line 5 assigns set $\{2, 6, 9, 12\}$ to site 6 and $\{2, 7, 10, 13\}$ to site 7.
- Line 6 assigns the jump-X diagonals as follows: Sets $\{3, 5, 9, 13\}$ to site 9, $\{3, 6, 10, 11\}$ to site 10 and $\{4, 7, 8, 12\}$ to site 8. Sets $\{4, 5, 9, 12\}$ to site 12, $\{4, 6, 8, 13\}$ to site 13 and $\{4, 7, 9, 11\}$ to site 11.

4. DISTRIBUTED ROUTING APPROACH

A major contribution of this paper is the design of a distributed routing, which achieves an efficient routing performance by considerably reducing the routing table size that needs to be maintained by each node. We assume initially that the original network is structured according to the design described in section 2. In this case, if a site needs to establish a path to a destination node located in its communication set then a direct connection can take place. However, if a site needs to establish a path to a destination node not in its communication set, then only one intermediate hop is needed to establish the path. In both cases, the maximum path size between any source and destination pair is composed of two hops. This can be illustrated through the following example. Given a network of size $N = 13$. Following our network topology design with communication sets $N_1 = \{1, 2, 3, 4\}$ and $N_6 = \{2, 6, 9, 12\}$ for sites 1 and 6 respectively. In this case, site 1 can directly establish a path to sites 2, 3, and 4. However in order to establish a path from site 1 to site 6, site one needs to go to the intermediate node 2, which represents the intersection of both communication sets. This design ensures a minimum routing path size.

As the network size gets larger, storing huge routing tables becomes a non desirable feature since it will consume a lot of the available storage resources as well as makes it difficult to maintain the routing tables. With our distributed approach, there is no need for the network nodes to store all network topology information. Our proposed idea is as follows: each node will maintain and store just its communication set. If a request for a path establishment between a source s and a destination d in the communication set comes in, then a direct communication can take place. However, in case a path request for a destination not in the communication set of the source node, then only the communication set of the destination set is generated on the fly and a route will be established from the source to the destination through the intersection node of the communication sets.

Following this approach, in order to route from a source to a destination site, any site needs to know only the communication set of the destination site. In this case, the intersection of the source communication set with the destination communication set represents the intermediate hop that the requested path should use. Algorithm 5 is used to achieve this goal; it takes as input the destination site number and the communication set size (K).

Figure 5 illustrates algorithm 5. It is important to notice that the generation of the communication sets depends only on the site number. Line 3 of the algorithm finds the index of the destination node n in the matrix. If the communication set to be generated is for node 1 or node 2, then the requested communication set S consists of the first row or the first column of the matrix. The complexity of finding the index (i, j) is $O(1)$. Generating the communication sets has order of complexity $O(K)$. Therefore, the overall complexity of the algorithm is $O(K)$.

5. RESULTS

We compare our design to a fully connected network. We consider as network metrics: wiring costs, routing table size, communication cost, and network quality. The following section provides the comparison results.

5.1 Routing Table Size

Algorithm 5: Generating the communication set of the destination node

```
Data: Destination node (n), |K|
Result: Communication set of the destination node (S)
begin
    generate matrix M
    find index (i', j') for site n in M
    if (n == 1) then
        S = M(i, j), i = 0, j = 0 → K
    else
        if (n == 2) then
            S = M(i, j), j = 0, i = 0 → K
        else
            if j' == 0 then
                S = M(i, j), i = i', j = 0 → K
            else
                Depending on i', find the jump;
                S = M(i, j), i = 0 → K, i = i * jump,
                j = 0 → K
end
```

Usually, an $(N \times N)$ routing table is stored in every node in order to make the routing decisions. Using the suggested topology, no routing table needs to be stored on any one of the nodes. The route is computed on the fly (on-demand).

5.2 Wiring Cost

The number of wires in a completely connected network $= \frac{N(N-1)}{2}$ while the number of wires in the described topology $= N(K-1)$. This is due to the fact that every site is connected to the $K-1$ sites in its communication set.

$$Gain = \frac{\sqrt{N}+1}{2} \tag{4}$$

5.3 Communication Cost

The number of hops used in a completely connected network is $N(N-1)$ because each node can reach every other node using one hop. In the suggested topology each node can communicate with other nodes by either 1 or 2 hops. So, the total number of hops is $N\times$ (no. of 1 hop + 2(no. of 2 hops))$= N\times(2(K-1)+2(K-1)(K-2)) = 2\times N\times(\sqrt{N}-1)^2$.

$$Loss = 2 \times \frac{\sqrt{N}-1}{\sqrt{N}+1} \tag{5}$$

5.4 Network Quality

The quality of a network is measured using the following equation:

$$Quality = \frac{1}{d \times D \times L} \tag{6}$$

, where d is the node degree, D is the network diameter, and L is the number of links. In a completely connected network: $d = (N-1)$, $D = 1$, and $L = N(N-1)/2$. Therefore,

$$Q1 = \frac{2}{(N-1)^2 \times N} \tag{7}$$

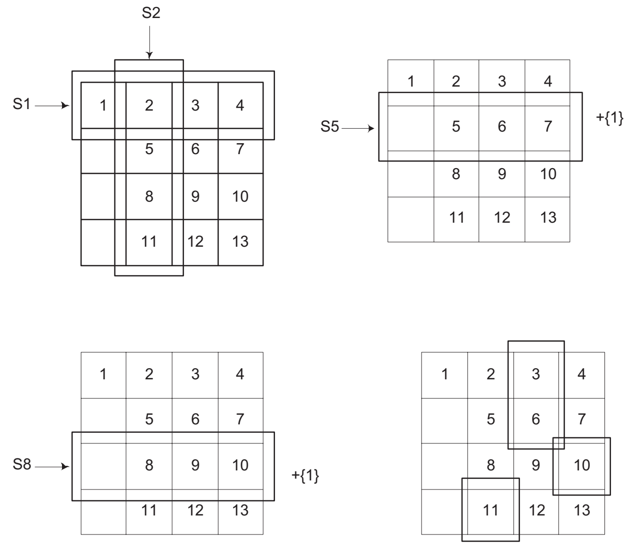

Figure 5: Automatic set generation

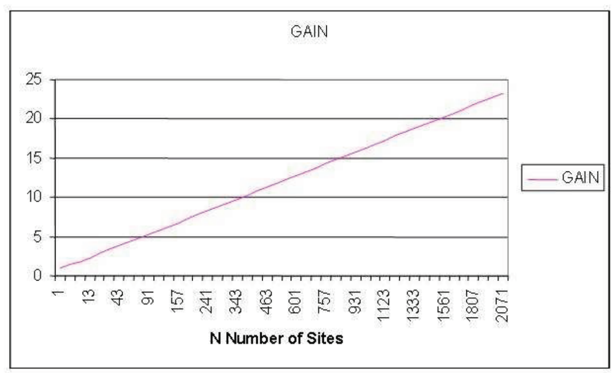

Figure 6: Gain increases linearly

In the suggested topology: $d = 2 * (K-1)$, $D = 2$, and $L = 2 * N * (K-1)$

$$Q2 = \frac{1}{8 \times N \times (\sqrt{N}-1)^2} \tag{8}$$

5.5 Analysis

It can be clearly seen that the gain (equation 4) is much bigger from the loss (equation 5). As N increases, the gain increases in an almost linear fashion (figure 6), while the loss is bounded by a constant (figure 7). Moreover, the quality of the suggested network topology is better than that of the completely connected network.

6. CONCLUSION

Routing protocols are considered as one of the major components in a distributed environment. Having a routing protocol with a good performance is very important especially with increasing network sizes as well as increasing demands for high QoS. In the first part of the paper we proposed a new network topology design with minimum number of links. The resultant network is resilient to network dynamics and has high quality. In the second part of the paper we

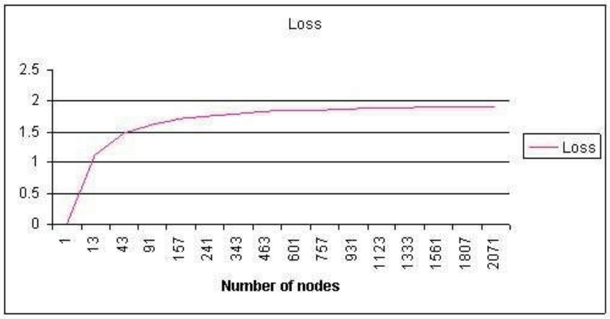

Figure 7: Loss converges to a constant

proposed a distributed routing protocol. We illustrated by simulation that this protocol reduces both communication overhead and storage space. We mentioned in the introduction that in case the ad hoc network is already deployed we will propose a way to map our suggested topology to the already existing one. We decided to keep the mapping part as future work.

7. REFERENCES

[1] C. P. Low and Y. J. Lee. Distributed multicast routing with end-to-end delay and delay variation constraints. *Computer Communications*, 24(9):848–862, 2000.

[2] Perkins, C. E., Royer, E. M., and Das, S. R. Ad hoc on-demand distance vector routing. *IETF Draft*, October 1999.

[3] Johnson, D., and Maltiz, D. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, T. Imelinsky and H. Korth, Eds. Kluwer Academic Publishers, pages 153–181, 1996.

[4] Royer, E. M., AND Toh, C. K. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55, 1999.

[5] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. *Proc. IEEE INFOCOM*, 3:1405–1413, Japan, April 1997.

[6] TOH, C.-K. A novel distributed routing protocol to support ad hoc mobile computing. *Proc. 1996 IEEE 15th Annual Int'l. Phoenix Conf. Comp. and Commun.*, pages 480–486, March 1996.

[7] TOH, C.-K. Long-lived ad-hoc routing based on the concept of associativity. *IETF Internet Draft*, March 1999.

[8] M. Maekawa. A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems. In ACM Trans. *Computer Systems*, 3(2):145–159, May 1985.

[9] J. Wu. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. *Journal of Communications and Networks*, 4(1):59–70, March 2002.

[10] J. Shaikh, J. Solano, I. Stojmenovic, and J. Wu. New metrics for dominating set based energy efficient activity scheduling in ad hoc networks. *Proc. of WLN Workshop (in conjunction to IEEE Conference on Local Computer Networks)*, pages 726–735, October 2003.

[11] J. Wu, M. Gao, and I. Stojmenovic. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. *Proc. of International Conference on Parallel Processing* (ICPP), pages 346–356, 2001.

[12] J. Wu and B. Wu. A transmission range reduction scheme for power-aware broadcasting in ad hoc networks using connected dominating sets. *Proc. of 2003 IEEE Semiannual Vehicular Technology Conference* (VTC2003-fall), October 2003.

[13] J. Wu, B. Wu, and I. Stojmenovic. Power-aware broadcasting and activity scheduling in ad hoc wireless networks using connected dominating sets. *Proc. of IASTED International Conference on Wireless and Optical Communication* (WOC 2002), 2002.

[14] J. Wu, B. Wu, and I. Stojmenovic. Power-aware broadcasting and activity scheduling in ad hoc wireless networks using connected dominating sets. *Wireless Communications and Mobile Computing, a special issue on Research in Ad Hoc Networking, Smart Sensing, and Pervasive Computing*, 3(4):425–438, June 2003.