November 2002

Routing in mobile ad hoc networks with global knowledge

D. Platt
University of Wollongong, dplatt@uow.edu.au

Routing in mobile ad hoc networks with global knowledge

Abstract

This paper describes a method of routing in mobile ad hoc networks. The problem is exacerbated by the fact that the radio links between the nodes of the network are continually being made and broken. The method consists of using the known connections to construct a "map" of the network which is intended to reproduce the geographical layout of the network at any time. An algorithm is described which finds a route from one node to another by always attempting to move geographically closer to the destination at every hop. This produces nearly optimal performance with very short computation time. The signaling load on the system is quite low. The scaling properties of the system are investigated and it is found that the system is viable with nodes numbering in the hundreds.

Keywords

ad hoc networks, mobile radio, telecommunication signalling

Disciplines

Physical Sciences and Mathematics

Publication Details

This paper originally appeared as: Platt, D, Routing in mobile ad hoc networks with global knowledge, The 8th International Conference on Communication Systems, 25-28 November 2002, vol 2, 1035-104.

ROUTING IN MOBILE AD HOC NETWORKS WITH GLOBAL KNOWLEDGE

Don Platt

University of Wollongong, Northfileds Avenue, Wollongong, NSW, Australia Email: d.platt@uow.edu.au

Abstract - This paper describes a method of routing in mobile ad hoc networks. The problem is exacerbated by the fact that the radio links between the nodes of the network are continually being made and broken. The method consists of using the known connections to construct a "map" of the network which is intended to reproduce the geographical layout of the network at any time. An algorithm is described which finds a route from one node to another by always attempting to move geographically closer to the destination at every hop. This produces nearly optimal performance with very short computation time. The signaling load on the system is quite low. The scaling properties of the system are investigated and it is found that the system is viable with nodes numbering in the hundreds.

Keywords – ad hoc networks, routing

List of Symbols

N = number of nodes
r = radio range, m
A = area of network, m^2
v = average speed of nodes, m/sec
C = number of nodes used to compute routes
n = average number of links in a route
τ = average duration of calls
t_u = map update time

I. INTRODUCTION

Mobile ad hoc networks consist of a collection of mobile nodes and the radio links between them. Each node acts as both a source/destination for messages and as a switching or routing node. There are no large scale switches or routers involved. In this paper, it will also be assumed that the network is flat, that is, there is no hierarchical structure; all nodes are essentially equal.

The mobile nature of the nodes ensures that radio links will be continually made and broken, as nodes move into and out of each other's radio range. This places a significant challenge on the network, since routes are always being disrupted, even after a call has been set up. The faster the nodes are moving, and the more calls are in progress, the more this is a problem.

However, the routing problem itself, may be much less demanding than it is in a fixed network. At any time, a route must be found from one node to only one other node, unlike the case of a fixed network, where routes must be found from every edge node to every other edge node.

This paper will address the problem of routing in a mobile ad hoc network. The approaches that have been explored in the literature may be classified in different ways. One classification is based on whether the routing scheme is "proactive" or "reactive".

In proactive schemes, the network is always pre-computing the best routes for every source/destination pair. This is standard in fixed networks, where such algorithms as Dijkstra's and Bellman-Ford may be used to compute the optimal routes from a source node to all other nodes in the network. In a mobile ad hoc network, there are a number of serious difficulties with this approach. Firstly, it supplies the network with more information than it needs, since a route will be required from a source node to only one destination node at any time. So memory resources may be used for no good purpose. Secondly, the number of nodes is likely to run into the hundreds, and the time taken to run the Dijkstra algorithm becomes prohibitive with such numbers. And thirdly, links are always being made and broken in a mobile ad hoc network, necessitating continual updating of the configuration information to all nodes involved in route computation, and rendering the existing information out-of-date and possibly misleading. These disadvantages have weighed heavily against proactive schemes, except in cases where there are few nodes. These schemes tend to require significant computation resources, but relatively little signaling bandwidth.

In reactive schemes, routes are generated, as they are needed. Since no pre-existing route is available, there may be a significant response time to a request for a connection. The simplest of these consists of flooding the network with search packets, which remember the route they have taken until finally, one or more search packets reach the destination. The route is then established and follows the best of the paths taken by the successful search packets. This requires very little computing power or memory, but uses a great deal of bandwidth.

Much recent work studies various hybrids of the two methods above. Schemes have been advanced which use some stored knowledge of the network, but not the complete configuration, and using search packets, but directing them in a more efficient manner. A selection of approaches is briefly described below, but more detail can be found in [1].

0-7803-7510-6/02/$17.00 ©2002 IEEE

Zone routing is a method that requires each node to have a detailed knowledge of the connections to nodes in its immediate vicinity, which is called its zone. For a node to set up a connection to another node, it first searches its own zone. If it fails to find the destination node, it must then send search packets out to the zones on the edge of its own zone. A search then follows within each of these zones, and the process continues until the destination is found. This method is proactive within each zone, but reactive between zones. Zone routing is discussed in [2, 3, 4, 5].

In [6] the authors propose two routing procedures which have similarities to zone routing. In hierarchical state routing, the members of a flat network are assigned to clusters, each of which has an elected node as its head. Among the cluster heads, the same procedure is carried out, until a virtual hierarchy has been formed. Routing proceeds by each cluster head examining the membership of its cluster for the required destination node. If it cannot find it, it refers the matter to the cluster head above it. This proceeds until the destination node is found, after which we go down the hierarchy of clusters to the destination. By the time this process is over, the route has been generated.

The so-called "fisheye" state routing [6] is inspired by the image produced by the eye of a fish, which provides fine detail close to the focal point but less detail as we move away from it. Because each node only requires detailed information of the region immediately adjacent to it, the amount of updating information is much less than the requirements of a fully proactive system.

In [7] the authors use a subset of the network nodes, called the "core", to perform all routing calculations. Every node in the network is connected to at least one core node, and membership of the core changes as the configuration changes. Routes are generated by a broadcast, which is intended only for members of the core. Thus this method is a reactive method with a reduced number of nodes being used to find destination nodes.

Similar in their approach are [8] which uses a "backbone" subnetwork for all signaling and [9] which uses a "spine" to propagate topology changes.

This paper will approach the problem with the intention of computing the routes with global knowledge of the network. A map of the network will be maintained, and to this extent, the method is proactive. However, a route will be computed only on demand, and this is typical of reactive methods.

II. PROPOSED ROUTING PROCEDURE

It is proposed that all routing calculations be carried out with all the current available information on the configuration of the network. In principle, there would be a single node that performed all these calculations, instantaneously, as required. In practice, the computation burden will prove to be so great that it has to be shared among a number of nodes that are designated for the task. Each of these route-calculating nodes has a list of every node in the network and all the nodes to which it is connected (via a radio link).

From this information, the calculation follows two distinct steps. The first step is to generate a map of the area covered by the nodes. The intention is for this map to resemble the actual geographical layout of the nodes as closely as possible. The second step is to generate a path from the calling node to the receiving node. This step is facilitated by the information, contained in the map, on the geographical positions of the two nodes concerned, and all the intermediate nodes.

III. CONSTRUCTION OF THE MAP

It is necessary to start with three nodes which all have radio contact with each other. In the new map, they will form an equilateral triangle with sides of unit length. We will build the map out from this triangle by adding the other nodes in rings of steadily increasing radius. The radius of each ring will be one unit greater than the radius of the previous ring.

The map is constructed as follows.

1. Start with an arbitrary node, (say node 1) and look for a set of three nodes, which are connected to each other.
2. Place these three nodes at points $(1/\sqrt{3}, 0)$, $(-1/2\sqrt{3}, 0.5)$ and $(-1/2\sqrt{3}, -0.5)$ on a plane. These three points are the apexes of an equilateral triangle with sides of unit length. These three nodes form the inner ring.
3. Each node, connected to all points on the previous ring, is allocated a position at the origin.
4. We form a new ring as follows. Each new ring has a radius greater than the previous ring by one. Each node, connected to one point only on the previous ring, is allocated a position on the new ring, on the same radius (ray) as the point to which it is connected, on the old ring.
5. Each node, connected to two points on the previous ring, is allocated a position on the new ring, halfway between the radii passing through the points to which it is connected, on the old ring.
6. And so on for nodes connected to more nodes on the old ring.
7. Return to 4. and continue until all nodes are located.

This produces a map in which all nodes are placed near to those to which they are connected. The orientation of the map is arbitrary. Also, it places nodes at the same point, if they have a radio connection to the same node(s) on the ring immediately inside them. Figure 1 (above) shows a set of 100 nodes distributed through a square of dimension 1000x1000 m^2 and the radio links are shown for a range of 150 m. Two

nodes are assumed to be in range if the distance between them is less than 150 m.

Figure 1 (below) shows the map produced by the procedure described above. Where nodes have been placed at the same point in the map, they have been moved a small distance in Figure 1 (below) so that there is more than one node visible at such points. It is clear that the map produced by this procedure does not look a great deal like the original geographical map. However, it will be found to allow reasonably economical routing.

IV. ROUTING ALGORITHM

The routing procedure may begin once the map has been produced. The algorithm is designed so that, at each step, it moves as close as possible to the destination node. After every hop, it looks at the positions of all the unused nodes connected to the latest node in the path. It chooses the node closest to the destination. If it becomes "cliff-bound", it backtracks and starts looking again. It does not consider nodes, which it has already tried and rejected.

The algorithm proceeds as follows.

1. A starting and a finishing node are specified. The path is started from the starting node. The "current node" is the starting node to begin.
2. All the nodes connected to the current node are tested for closeness to the finishing node, and the closest one is chosen to be the new "current node". It is added to the route. The new "current node" is also flagged as having been used, and is ineligible for consideration at a later stage of the routing process.
3. If no node is eligible, then the current node is removed from the route and the algorithm backtracks by going back to the previous node in the route, and this becomes the new "current node".
4. Return to 2. until we reach the destination, or there are no more eligible nodes connected to the "current node"

This algorithm is not guaranteed to find the route with the smallest number of hops, but it will always find a route if one exists. It is, of course, possible that a mobile ad hoc network may not have a route between two particular nodes. In this case, the algorithm finishes on the starting node, having exhausted all possibilities.

Figure 1 shows three routes (in solid lines) that have been calculated by this method.

V. PERFORMANCE OF ALGORITHM

The most important factors affecting the performance of the routing algorithm are the number of nodes in the network, the area covered by the network and the radio range. In general, it is more difficult to find a route through a network if the area covered is large, and if the radio range is small. If the area is densely populated with nodes, then this makes it easier to find routes through the network. As the area increases, and the range decreases, we may expect the number of hops to increase.

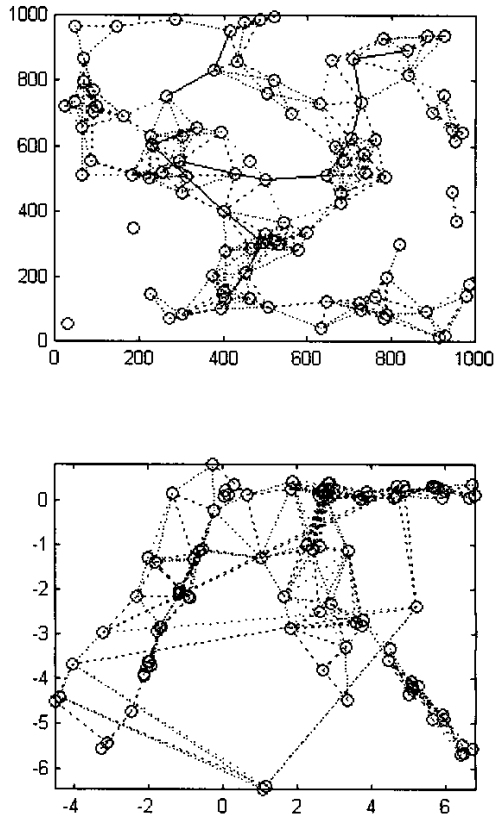

Figure 1. Radio Connections and Routes Spatial Distribution of Nodes (above) Reconstructed Map (below)

If this is taken too far, then the network may break up into separate "islands", and then it is not possible to find routes between nodes located in different islands. If the area is relatively small and the range relatively large, then it becomes a much easier task to find a route, and the number of hops

becomes much smaller, and the route may frequently consist of a single hop.

Our interest will lie between these two extremes, where the network is more or less free of islands, but several hops are necessary to connect most pairs of nodes.

The performance of the algorithm will be measured by the number of hops required to connect two nodes, compared to the minimum possible number of hops, as computed by Dijkstra's algorithm. It will also be measured by the number of floating point operations (FLOPs), and the computation time required to generate the map and to compute the route.

Since the routing algorithm is not guaranteed to find the route with the lowest number of hops, it is worthwhile to inquire whether the same algorithm might find a better route by starting from the other end. This has the advantage of offering two routes, one of which will quite likely involve a lower number of hops than the other. However, it has the disadvantage that it will double the computation time, on the average.

The tables below show the results of running a series of simulations for an ad hoc network with the number of nodes increasing geometrically from ten to one thousand. In all cases, the radio range was set to 150 m and the area covered by the network was set to the number of nodes times 10,000 square metres. The area was made up of a square of appropriate dimensions, and the nodes were scattered through the area with uniform probability of occurring at any particular location. Every simulation consisted of

(a) generation of the map as used by the route computing nodes,

(b) generation of a route between two randomly selected nodes, in both directions, using the map from (a),

(c) generation of a route between the same two nodes, in both directions, using the accurately known position of all nodes, and

(d) generation of the route with the fewest hops, using the Dijkstra algorithm.

The tests were run 50 times and the averages were taken. Table 1 gives the average number of hops found in the various paths. Table 2 gives the computation time as reported by the Matlab program suite, using a desk-top computer with a Pentium processor running at a clock speed of 350 MHz. Table 3 gives the corresponding number of FLOPs.

The number of hops increases with the number of nodes in the network, as would be expected. The number of hops increases in proportion to $N^{0.49}$ using the Dijkstra algorithm, and $N^{0.51}$ with the new algorithm applied in both directions to a map with accurate positions. Using the accurate map, but only calculating the path in one direction produced a number of hops proportional to $N^{0.57}$. When the new algorithm is applied, in both directions, to the map generated internally, the number of hops varies with $N^{0.68}$, and when it is applied in one direction only, it varies with $N^{0.82}$.

Table 1. Number of Hops Using Various Methods for Different Sized Networks

Nodes	Hops Using Map	Using Map Both Ways	Hops Using Accurate Position	Using Accurate Position Both Ways	Hops Using Dijkstra
10	1.78	1.73	1.73	1.73	1.73
32	4.20	3.50	3.89	3.54	3.29
100	12.3	7.49	6.44	5.65	5.41
316	31.4	20.9	12.9	10.1	8.92
1000	79.7	42.6	25.7	18.3	15.0

Two conclusions can be drawn from these results. Firstly, whether the new algorithm is applied using the map generated internally or an accurate map, there is a significant improvement when it is used in both directions. Secondly, there is very little difference between the number of hops generated using the new algorithm in both directions and an accurate map, and the number of hops in an optimum path. So the new algorithm produces paths which are close to optimum, if it has an accurate map. However, the performance deteriorates significantly when the map used is the one generated internally from the information available on the connections. This suggests that there is more improvement to be gained by improving the map, rather than by improving the route finding algorithm.

The computation time, required to generate the map, is proportional to $N^{1.27}$, and the number of FLOPs is proportional to $N^{1.25}$.

The computation time, required to find a route using the internally generated map, is proportional to $N^{0.93}$, and the number of FLOPs is proportional to $N^{0.89}$. And to find a route using an accurate map, the corresponding functions are $N^{0.79}$ and $N^{0.67}$. Again, there is a significant advantage available if it is possible to use an accurate map.

Although it is not intended to propose the use of the Dijkstra algorithm here, it is interesting to note that the computation time is proportional to $N^{1.46}$ and the number of FLOPs is proportional to $N^{1.45}$. The Dijkstra algorithm is

much more powerful than what is needed here, computing the optimal route from the source node to all other nodes, whereas we only need to find a route to one other node. It also requires much more computation time.

Table 2. Computation Times in Seconds

Nodes	Generate Map	Find Route in Map	Find Route with Accurate Pos'n	Find Route Using Dijkstra
10	0.0068	0.0022	0.0018	0.0888
32	0.0257	0.0048	0.0040	0.3764
100	0.0950	0.0141	0.0080	1.6170
316	0.4295	0.0456	0.0214	8.1971
1000	2.1087	0.1474	0.0607	74.6785

Table 3. Average Number of Floating Point Operations (FLOPs)

Nodes	Generate Map	Find Route in Map	Find Route with Accurate Pos'n	Find Route Using Dijkstra Algorithm
10	10	150	90	220
32	630	230	210	1,150
100	2,470	640	350	5,680
316	11,230	1,920	790	29,910
1000	50,860	5,590	1,900	162,860

Tables 4, 5 and 6 show the results of running the same set of tests on networks covering half the area. The average number of hops in the routes is reduced approximately by a factor of $1/\sqrt{2}$. The time required to generate the map still scales with $N^{1.27}$, and the number of FLOPs is still proportional to $N^{1.26}$. Under these conditions, the scaling improves, so that the computation time to find a route using the internally generated map now scales with $N^{0.8}$, and the number of FLOPs is also proportional to $N^{0.8}$. To find a route using an accurate map, the scaling functions are $N^{0.63}$ and $N^{0.57}$. The scaling is similar, but where there is a difference, it is found that the scaling properties are improved when the network is distributed over the smaller area.

Table 4. Number of Hops Using Various Methods for Different Sized Networks over Reduced Area

Nodes	Hops Using Map	Using Map Both Ways	Hops Using Accurate Pos'n	Using Accurate Pos'n Both Ways	Hops Using Dijkstra
10	1.29	1.24	1.24	1.24	1.24
32	2.95	2.46	2.28	2.28	2.28
100	6.65	5.30	3.80	3.76	3.76
316	17.62	11.46	6.73	6.58	6.5
1000	34.27	18.1	10.78	10.64	10.38

Table 5. Computation Times in Seconds (Reduced Area Network)

Nodes	Generate Map	Find Route in Map	Find Route with Accurate Pos'n	Find Route Using Dijkstra
10	0.0058	0.0020	0.0022	0.0976
32	0.0273	0.0042	0.0030	0.5013
100	0.1015	0.0088	0.0065	2.4682
316	0.4025	0.0246	0.0126	13.525
1000	1.7297	0.0610	0.0267	111.90

VI. SCALING ISSUES

The various measures of the performance of a network change with the number of nodes. The results presented in the last section bring some of these out. Of particular significance is the average number of hops in routes through the network. Simple calculations suggest that the number of hops should be proportional to the linear dimension of the area covered, or $\sqrt{A}$, and that it should be inversely proportional to the radio range. The results show that the best which is physically possible, as computed by the Dijkstra algorithm is almost matched by the routing algorithm suggested here, throughout the range $10 < N < 1000$, if we can use an accurate physical map of the layout of the network. For these cases, the average number of hops is approximately $0.8\frac{\sqrt{A}}{r}$. When we use the

internally generated map, the scaling changes to $0.54\frac{\sqrt{A}N^{0.18}}{r}$ over the larger area and this improves to $0.68\frac{\sqrt{A}N^{0.1}}{r}$ when the network is distributed over half that area.

Table 6. Average Number of Floating Point Operations (FLOPs) (Reduced Area Network)

Nodes	Generate Map	Find Route in Map	Find Route with Accurate Pos'n	Find Route Using Dijkstra
10	130	80	80	310
32	600	230	190	2,060
100	2,350	520	340	11,400
316	9,360	1,350	640	64,480
1000	38,000	2,590	1,020	369,940

The time taken to compute the routes depends on the length of the route, but also the likelihood of backtracking. This is clearly greater when the area is more sparsely populated with active nodes. Using the internal map, the computation time is equal to $0.00022*N^{0.93}$ when the large area is used, and it is equal to $0.00025*N^{0.80}$ when the area is halved. Using the accurate map, the computation times are $0.00025*N^{0.79}$ and $0.00036*N^{0.63}$, respectively. The number of FLOPs follows the computation times fairly closely. This is reasonably good scaling behaviour.

The computation time required for generating the map does not scale so well, and is approximately $0.00031*N^{1.26}$ for both cases.

However, the aspect of the scheme, which fares worst, in terms of scaling, concerns the C route calculating nodes. For a complete update of the connection tables, C(C-1) packets have to be transmitted. Rough calculations suggest that C is proportional to N raised to a power between 2.5 and 3. (It is this aspect which limits the practical size to about 400 nodes, using the parameters of this study. When N = 690, C = 690.) While this has the benefit of reducing the payload of these packets, it suggests that the control traffic generated by this activity scales with something like N^4 or N^5.

VII. COMMENTS AND CONCLUSIONS

This paper has described a method for routing in a mobile ad hoc network. It describes a routing algorithm, based on the knowledge of the geographical location of the two nodes to be connected. Under normal circumstances, it will respond quickly to requests, and will provide fairly good routes. If an accurate map can be generated, for example, using a GPS system, then the behaviour of the system improves. Routes and route computation times are shorter, and no time need be devoted to the computation of a map. This will noticeably raise the number of nodes that could be accommodated in the network.

The levels of signaling and control traffic produced are quite low because all routes are calculated rather than found by search packets passing around the physical network. The low response time and control traffic are the main attractions of the system.

The burden of routing falls on computing resources rather than bandwidth, and it is here that the system runs into its limits. As computing speed and available memory continue to increase, these limits will be pushed back. For example, if the computing speed doubles, the necessary number of route computing nodes will halve, but the scaling properties attached to the number of necessary computing nodes do not offer easy increase in the size of the network.

The system looks to be very efficient with something like 100-200 nodes, requiring only 2 route computing nodes when N = 100 and a maximum of 12 route computing nodes when N = 200. It may be productive to limit the size of a flat network to these levels, and to design larger networks on a hierarchical principle, using the system described here as the lowest level.

REFERENCES

1. E. M. Royer and Chai-Keong Toh, "A review of current routing protocols for ad hoc mobile wireless networks", IEEE Personal Communications, vol 6, no 2, April 1999, pp 46-55

2. M. R. Pearlman and Z. J. Haas, "Determining the optimal configuration for the zone routing protocol," IEEE J. Select Areas Commun., vol 17, no 8, pp. 1395-1414, Aug. 1999.

3. M. Hoa-Ng and I.-T. Lu, "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks," IEEE J. Select Areas Commun., vol 17, no 8, pp. 1426-1438, Aug. 1999.

4. M. R. Pearlman, Z. J. Haas and S. I. Mir, "Using routing zones to support route maintenance in ad hoc networks",

IEEE Wireless Communications and Networking Conference, Chicago, Sept 2000, vol 3 pp 1280-1285
5. K. Shea, R. W. Ives and M. Tummala, "Mobile ad hoc network routing protocol analysis and its application to a programmable modular communication system", Proc 34 th Asilomar Conf on Signals, Systems and Computers, Pacific Grove, Oct-Nov 2000, vol 2, pp 1260-1264
6. A. Iwata, C.-C Chiang, G. Pei, M. Gerla, and T.-W. Chen, "Scalable routing strategies for ad hoc wireless networks," IEEE J. Select Areas Commun., vol 17, no 8, pp. 1369-1379, Aug. 1999.
7. R. Sivakumar, P. Sinha, and B. Bharghavan, "CEDAR: A core extraction distributed ad hoc routing algorithm," IEEE J. Select Areas Commun., vol 17, no 8, pp. 1454-1465, Aug. 1999.
8. B.Das, and V. Bharghavan, "Routing in ad hoc networks using minimum connected dominating sets", IEEE International Conference on Communications, 1997, Montreal, June 1997, pp 8-12
9. B. Das, R. Sivakumar and V. Bharghavan, "Routing in ad hoc networks using a spine", Proc. Sixth Int Conf on Computer Communications and Networks, 1997, Las Vegas, Sept 1997, pp 34-39