This research has been supported in part by European Commission
FP6 IYTE-Wireless Project (Contract No: 017442)

A Cluster Based Hierarchical Routing Protocol for Mobile Networks

Kayhan Erciyes and Geoffrey Marshall

California State University San Marcos,
Computer Science Dept., 333 S.Twin Oaks Valley Rd.,
San Marcos, CA 92096, U.S.A.
{kerciyes,marsh021}@csusm.edu

Abstract. We propose a hierarchical, cluster based semi-distributed routing protocol for a mobile ad hoc network. The network graph is partitioned into clusters using heuristics and the shortest routes are first calculated locally in each cluster in the first phase. The network is then simplified to consist only of the nodes that have connections to other clusters called the neighbor nodes and the shortest routes are calculated for this simple network in the second. A complete route between two nodes of different clusters is formed by the union of intra-cluster and inter-cluster routes. We show that this method has better performance with respect to the other methods of calculation of all-pairs shortest paths in a mobile network. ...

1 Introduction

Two general ways for building routing tables in an arbitrary computer network are the central and distributed approaches. In the central approach, connectivity information from all nodes are gathered in a central coordinator which performs some routing algorithm and distributes the routing tables to individual nodes. Dijkstra's *All-pairs Shortest Paths* (APSP) algorithm [2] uses the greedy approach and finds all routes in $O(n^3)$ time. Floyd and Warshal algorithm [4] uses dynamic programming and finds the routes similarly in $O(n^3)$ time. Distributed routing algorithms assume that there is no central component and global information available to the nodes of the network. Each node makes use of its local information and information it receives from its neighbors to find the shortest routes. Mobile ad hoc networks do not have central administration or fixed infrastructure and consist of mobile wireless nodes that have temporary interconnections to communicate over packet radios. The rapidly changing topology of a mobile network requires that routes should be calculated much more frequently than the wired networks. Distributed, adaptive and self-stabilizing algorithms may be used to perform routing in mobile networks. *Link reversal routing* algorithms are one class of such algorithms where a node reverses its incident links when it loses routes to the destination. Performance analysis of link reversal algorithms are given in [1] and TORA [9] is an example system that uses link reversal routing. An important routing approach in mobile networks is

clustering, that is, partitioning of the network into smaller subnetworks to limit the amount of routing information stored at individual nodes. In [8], a mobile network is partitioned into clusters of a two level graph. In the *zone routing* proposed in [5] where a zone functions similar to a cluster, the requested routes are first searched within the local zone. For inter-zone routes, the search is carried by multicast messages to the boundary nodes within the zones. In *k-way clustering*, the mobile network is divided into non-overlapping clusters where two nodes of a cluster are at most k hops away from each other. A k-way clustering method is proposed in [3] where the spanning tree of the network is constructed in the first phase and this tree is partitioned into subtrees with bounded diameters in the second phase.

In this study, we propose a hierarchical, semi-distributed, two-level dynamic distributed routing protocol for a mobile network. The protocol is not fully distributed due to the existence of some privileged nodes in the network. The distributed routing architecture consists of hierarchical clusters of routing nodes and each cluster has a controller which is called the *representative*. At the highest level, one of the representatives called the *coordinator* has the complete connectivity information of all the nodes in the network. Everytime there is an addition or deletion of a node to a cluster, the coordinator is informed to update its view. Upon such changes of configuration or periodically gathering of the changes, the coordinator starts a new configuration process by partitioning the network graph into new clusters. The nodes in the cluster including the nodes that have connections to other clusters are called the *neighbor nodes*. The coordinator chooses one of the neighbor nodes in each cluster as the cluster representative and sends the cluster connectivity information and neighbor connectivity information to the representative of such a group. Each representative then distributes the local connectivity information to all of the nodes in its group which concludes the first phase of the protocol. In the second phase, each node performs APSP routing within its cluster. This phase is concluded by calculating the distances between all pairs of nodes in the cluster including the neighbor nodes. In the third phase, only the neighbor nodes calculate all-pairs shortest path routes for the *neighborhood graph* which represents the simplified inter-cluster connectivity of the original network. Any route is then formed by the union of the route from the source node to its nearest neighbor, the shortest route between the source neighbor and the destination neighbor and the shortset route between the destination neighbor and the destination node.

The rest of the paper is organized as follows. The partitioning of the network is described in Section 2, the distributed route management is explained in Section 3 and an example network is detailed in Section 4. The performance analysis of the overal system is given in Section 5. Finally, the implementation carried so far is presented in Section 5 and future directions are outlined in the Conclusions section.

2 The Partitioning of the Network

The aim of any partitioning algorithm is to provide subgraphs such that the number of vertices in each partition is averaged and the number of edges cut between the partitions is minimum with a total minimum cost. Various partitioning algorithms for graphs for task scheduling and related problems exist. An arbitrary network can be constructed as an undirected connected graph $G = (V, E, w)$ where V is the set of routing nodes, E is the set of edges giving the cost of communication between the routing nodes and w: $E \rightarrow \Re$ is the set of weights associated with edges. *Multilevel partitioning* is performed by coarsening, partitioning and uncoarsening phases [6]. During the coarsening phase, a set of smaller graphs are obtained from the initial graph $G_i = (V_i, E_i, w_i)$ such that $|V_i| > |V_{i+1}|$. When graph G_{i+1} is to be constructed from graph G_i, a maximal matching $M_i \subset E_i$ is found and vertices that are incident on both edge of this matching are collapsed. The collapsing is performed as follows. If $u, v \in V_i$ are collapsed to form vertex $z \in V_{i+1}$, the total weight of vertices u and v become the weight of z, the edges incident on z is set equal to the union of the edges incident on u and v minus the edge (u, v). If there is an edge that is incident on both u and v, then the weight of this edge is set equal to the sum of the weights of these two edges. Vertices that are not incident on any edge of the matching are copied to G_{i+1}. In the maximal matching, vertices which are not neighbors are searched. In HEM, the vertices are visited in random order, but the collapsing is performed with the vertex that has the heaviest weight edge with the chosen vertex. Vertices are visited in random order and an adjacent vertice is chosen in random order as well in RM. During the succesive coarsening phases, the weights of vertices and edges increase. The coarsest graph can then be partitioned and further refinements can be achieved by suitable algorithms like Kernighen and Lin [7]. Finally, the partition of the coarsest graph is iteratively reformed back to the original graph by going through the graphs G_{k-1}, G_{k-2}, ...,G_1.

For the routing protocol, we propose a partitioning method called *Fixed Centered Partitioning* (FCP) where several fixed centers are chosen and the graph is then coarsened around these fixed centers by collapsing the heaviest edges around them iteratively. Different than [6], FCP does not have a matching phase, therefore iterations are much faster. Fig. 1 shows an example where a regular graph of 10 nodes is partitioned into three clusters. The initial fixed centers are encircled and the first collapsed neighbors are shown. The collapsing phases and the nodes collapsed at each iteration are depicted in Fig. 1.b and Fig. 1.c. The final partition has 3 partitions and a total cost of 13 for inter-partition edges. One problem with FCP is the initial allocation of the fixed centers. One possible solution is to choose the fixed centers randomly so that they are all at least some bounded distance from each other. The heuristic for the bound we used is $h = 2d \,/\, p$ where d is the diameter of the network and p is the number of partitions (clusters) to be formed.

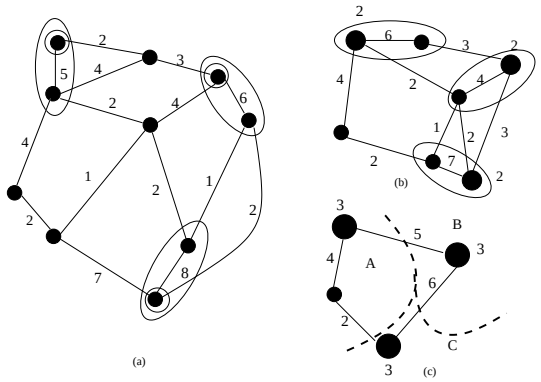

Fig. 1. Fixed Centered Partitioning

Lemma 1. *FCP performs partitioning of* $G(V, E)$ *in* $O(\lfloor n/p \rfloor)$ *steps where* $|V| = n$ *and* p *is the number of clusters (partitions) required. The time complexity of the total collapsing of FCP is* $O(n)$.

Proof. The FCP simply collapses p nodes with its heaviest edges at each step resulting in n/p steps. Since there are p collapsing at each step, total time complexity is $O(n)$.

3 The Hierarchical Routing Protocol

In the hierarchical routing protocol called the *Neighbor Protocol* (NP), each cluster has a representative neighbor node and one of the representatives is the coordinator. The distribution of the connectivity information is from the coordinator to the representatives and then from the representatives to the individual nodes as in a tree structure. When an ordinary mobile node changes its position, it sends a ND_TOP message to inform its representative that it now has different coordinates and goes into a WAIT state to wait until a new connectivity message (ND_ROUTE) is received from the representative as shown in Fig.2. The representative collects the ND_TOP messages until a timeout and then sends all of the current connectivity information to the coordinator in a REP_CONFIG message. The coordinator, after collecting the REP_CONFG messages within its timeout, starts the partitioning process of the updated network graph.

The coordinator concludes this step by identifying the nodes and neighbors in each cluster. It identifies one of the neighbor nodes as the representative for each cluster and sends the connectivity matrix (REP_PART) to the representatives. The representative for each cluster then distributes the local cluster connectivity information (ND_ROUTE) to individual nodes in its cluster in parallel with the other clusters . The representative also distributes the neighbor information to the neighbor nodes in its cluster. Ordinary nodes perform APSP to find their local shortest routes. Neighbor nodes however, perform APSP for intra-cluster and then inter-cluster routes.

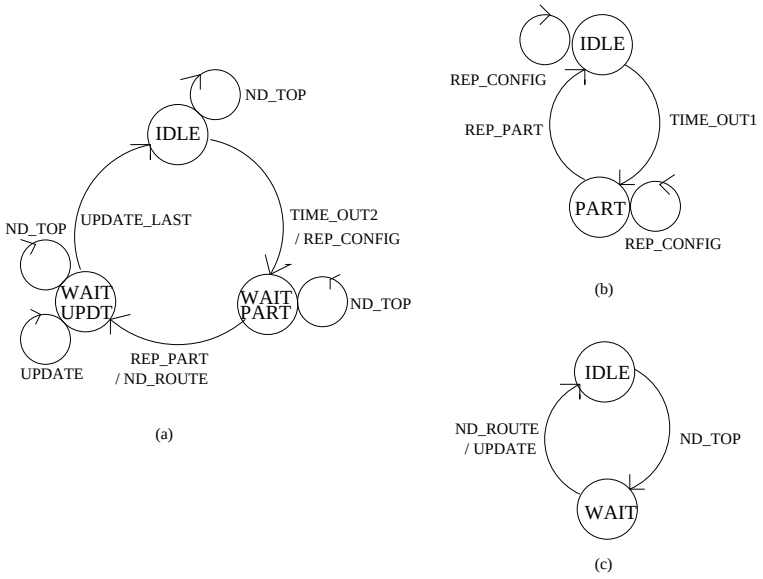

Fig. 2. The FSM Diagrams of Representative (a) , Coordinator (b) and Node (c)

4 An Example Network and Evaluations

An example network is depicted in Fig. 3. The initial centers allocated are 2, 11, 20 and 29. The coordinator and also the representative for cluster C is at 24. The coordinator partitions the graph using FCP as in Table. 1.

Table 1. Partitioning of the Example Network by FCP

A	B	C	D
G_0 — 2	11	20	29
G_1 — 2-1	11-12	20-19	29-28
G_2 — 2-1-5	11-12-8	20-19-17	29-28-25
G_3 — 2-1-5-4	11-12-8-14	20-19-17-22	29-28-25-26
G_4 — 2-1-5-4-3	11-12-8-14-15	20-19-17-22-21	29-28-25-26-23
G_5 — 2-1-5-4-3-6	11-12-8-14-15-13	20-19-17-22-21-18	29-28-25-26-23-27
G_5 — 2-1-5-4-3-6-7	11-12-8-14-15-13-10	20-19-17-22-21-18-16	29-28-25-26-23-27-24

Based on the partitioning information, the representatives chosen from the neighbors as 7, 10 and 17 are informed of their local connection. In the second phase, the representatives transfer this information to local nodes in their clusters in parallel. The ordinary nodes then calculate APSP in parallel, however, the

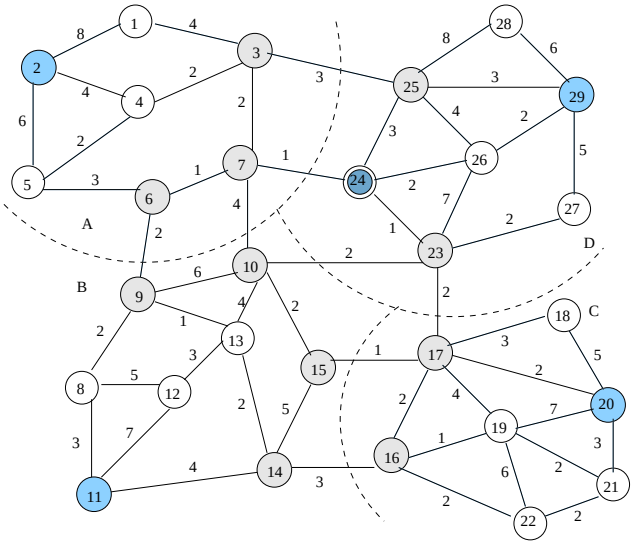

Fig. 3. The Original Network

neighbor nodes have to also calculate APSP for the simplified network graph which consists of the neighbor nodes only as shown in Fig. 4.

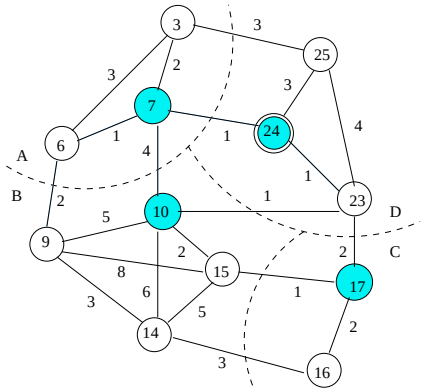

Fig. 4. The Simplified Neighbor Network

Consider an example where node 5 in cluster A wants to send a message to the node 22 in cluster C . Since destination is not in its own cluster, 5 sends the message to its closest neighbor node, 6. Node 6 sends the message to node 17 which is its closest neighbor node in cluster C over 6-7-24-23-17. The neighbor node 17 routes the message to the destination over the shortest path which is

17-16-22. The total cost of this path using NP or APSP is 12. The routes found by the method we proposed and any APSP algorithm such as Dijkstra's are compared for the example network for any pair of nodes that are in different clusters. The average coincidence for all pairs of nodes in each cluster for this example network, 85 % of the NP routes are coincident with APSP for a total of 308 routes.

5 Analysis

The performance analysis should include the following

1. Partitioning of the network graph by FCP : O_1
2. Distribution of the cluster connectivity messages to the cluster representatives : O_2
3. Distribution of the routing information to the individual nodes by each representative : O_3
4. Intra-cluster route calculation time by the nodes within the cluster : O_4
5. Inter-neighbor route calculation by the neighbors : O_5

Lemma 2. *Distribution of individual cluster routing information to the nodes (steps 2 and 3 above) take $O_{dist}(m)$ time where m is an upper bound on the number of nodes in a cluster*

Proof. The coordinator sends the cluster connections and the neighbor identities to all of the representatives in $\Theta_2(p)$ steps where p is the number of clusters in the network. Similarly, the representatives transfer this information to the individual nodes in $\Theta_3(m)$ steps in parallel with the other representatives. Assuming $m \gg p$, total time taken is $O_{dist}(m)$.

Lemma 3. *The total time required for intra-cluster and inter-neighbor routing algorithms is $O_{route}(m^3)$*

Proof. Since each node performs all-to-all routing in parallel, time spent for finding intra cluster routes is $O_4(m^3)$. Similarly, time spent for inter-neighbor shortest paths is $O_5(k^3)$ where k is an upper bund on the number of neighbors. Assuming $m \gg k$, total time is dominated by $O_{route}(m^3)$.

Theorem 1. *The Speedup obtained by the proposed protocol to a pure sequential all-to-all shortest paths protocol is $O(p^3)$ and to the parallel case where each node calculates all of the routes in parallel with others is $O(p^2/m)$.*

Proof. Total time for the protocol (O_{prot}) by Lemmas 1-3 is :

$$O_{prot} = O_{part}(n) + O_{dist}(m) + O_{route}(m^3) = O(n + m^3) \tag{1}$$

and assuming a balanced partition, that is, $n = mp$

$$O_{prot} = O(n + m^3) = O(mp + m^3) \tag{2}$$

Assuming the network has p clusters and m nodes at each cluster, a serial algorithm to compute all routes of this network will take $O_{serial}((p*m)^3)$ operations. The speedup S that can be approximated with respect to pure serial case is :

$$S = O_{serial}/O_{prot} = O((p*m)^3/(mp+m^3)) \tag{3}$$

and assuming $m \gg p$

$$S = O(p^3) \tag{4}$$

For the pure parallel case where each node has all of the network connectivity information, $O_{par} = O_(p^2m^2)$ and the speedup now is :

$$S = O(p^2m^2/m^3) = O(p^2/m) \tag{5}$$

This result may be interpreted as the more partitions the network has, the more speedup we obtain as $S = O(p)$ when $p = m$. This is not necessarily true as the number of partitions increases, the average nodes in a partition (m) would decrease for a given network which means that the assumption made about the relative magnitudes of p and m $(m \gg p)$ will not hold.

6 Experimental Results

The results for the partitioning of the graphs are shown in the figures below for the four algorithms as FCP, RFCP, RM and HEM where centers are initially allocated at random in RFCP. In Fig. 5, the average edge costs on sample graphs by the four algorithms are plotted. FCP and RFCP provide less total edge costs than the other two algorithms as shown.

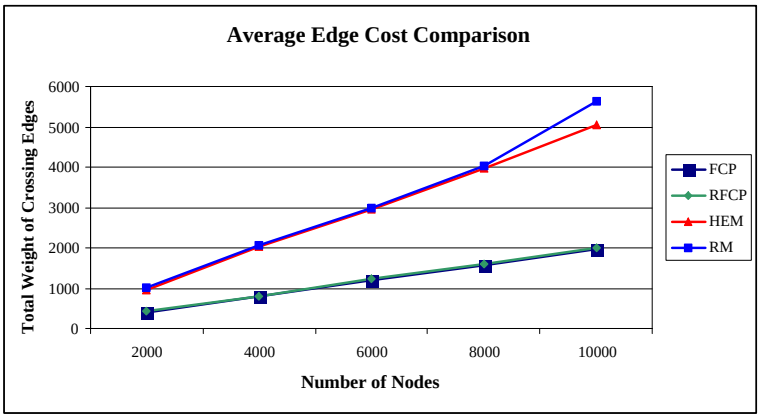

Fig. 5. Edge Cost Comparison

The partition quality of the FCP, RM and HEM algorithms are shown in Fig. 6 for 10000 nodes where FCP and RFCP both have significant improvements

over RM and HEM as expected since FCP partitions the graph into almost equal partitions as stated in Lemma 1.

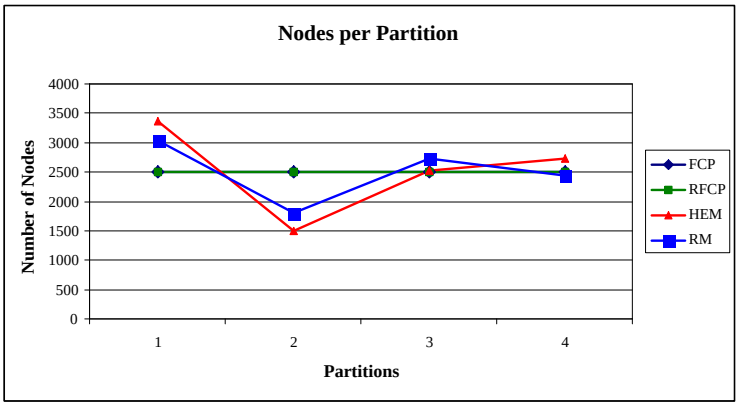

Fig. 6. Partition Quality Comparison

For the static routing tests shown in Fig. 7, the shortest routes were first calculated by APSP algorithm for various random sized graphs. Then, the graphs were partitioned into clusters, the neighbor graphs were constructed and the shortest paths were calculated using the Neighbor Protocol. The latter routes are compared to the APSP routes and their deviations are calculated. We see that NP has about 27 % deviation from APSP in the largest graph set under consideration which has about 10000 nodes.

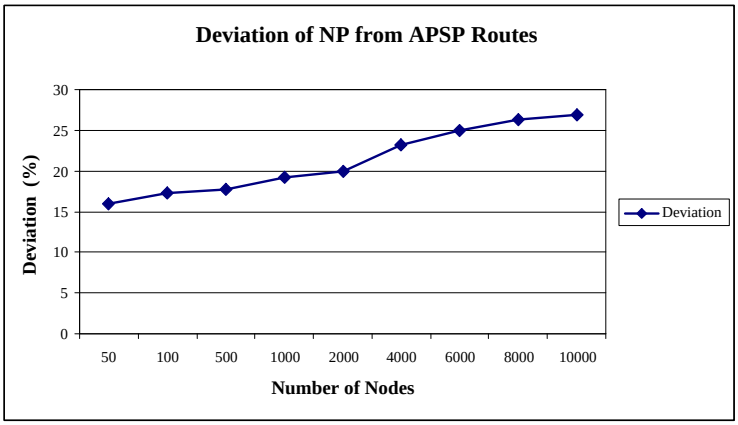

Fig. 7. Deviation of the Neighbor Protocol from APSP

7 Conclusions

We proposed a dynamic routing protocol for a mobile network called the Neighbor Protocol, provided a graph partitioning heuristic and a hierarchical model to perform routing in parallel. We showed that this approach improves performance considerably theoretically. However, although realistic, the speedup obtained in Section 5 is optimistic in a sense because of the few assumptions made about the number of clusters and the nodes in a cluster. The method we propose for routing in mobile networks provides *good* routes which are not necessarily the shortest paths but are comparable to shortest paths as shown by the tests. We are planning to evaluate the performance of NP in terms of total control traffic against the frequency of route requests and frequency of movement in a mobile network using simulations and comparing these with other mobile network routing protocols such as zone routing and k-way clustering.

We are also looking into the fully distributed version of this protocol for mobile ad hoc networks for two cases. In the first case, there is no central coordinator but there are representatives and decisions on the partitioning of the graph and routing are done at the representative level by distributed agreement. In the second case, there are no central components which would require fully distributed algorithms.

Acknowledgements

We wish to thank Pinar Dűndar of Ege University, Dept. of Math. for her valuable discussions of the graph partitioning heuristics.

References

1. Busch, C. et al.: Analysis of Link Reversal Routing Algorithms for Mobile Ad hoc Networks. Proc. of the 15th ACM Symp. on Parallel Alg. and Arch. (2003) 210-219
2. E.W. Dijkstra: A Note on Two Problems in Connection with Graphs. Numerische Math., Vol. 1. (1959) 69-271
3. Fernandess,Y., Malki, D.: K-clustering in Wireless Ad hoc Networks, Proc. of the second ACM Int. Workshop on Principles of Mobile Computing, (2002) 31-37
4. R.W. Floyd: Algorithm 97 (Shortest path), CACM, Vol. 5(6). (1962) 345
5. Z.J. Haas, M.R. Pearlman: The Zone Routing Protocol (ZRP) for Ad hoc Networks. Internet Draft, Internet Engineering Task Force. (1997)
6. Karypis, G., Kumar V.: Multilevel k-way Partitioning scheme for Irregular Graphs. Journal of Parallel and Distributed Computing, Vol. 48. (1998) 96-129
7. Kernighan, B., Lin, S.: An Effective Heuristic Procedure for Partitioning Graphs. The Bell System Technical Journal, (1970) 291-308
8. P. Krishna et al.: A Cluster-based Approach for Routing in Dynamic Networks. ACM SIGCOMM Comp. Comm. Rev., Vol. 27(2). (1997) 49 - 64
9. V.D. Park, M.S. Corson: A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. Proc. IEEE INFOCOM, Vol. 3. (1997) 1405-1413