Providing Full Connectivity in Large Ad-Hoc Networks by Dynamic Placement of Aerial Platforms

Karthikeyan Chandrashekar and Majid Raissi Dekhordi and John S. Baras
Institute for Systems Research
University of Maryland
College Park, Maryland 20742
Email: karthikc@isr.umd.edu , majid@isr.umd.edu , baras@isr.umd.edu

Abstract: In this paper we address the problem of providing full connectivity to disconnected ground MANET nodes by dynamically placing unmanned aerial vehicles (UAVs) to act as relay nodes. We provide a heuristic algorithm to find the minimal number of such aerial vehicles required to provide full connectivity and find the corresponding locations for these aerial platforms (UAVs). We also track the movement of the ground nodes and update the location of the UAVs. We describe a communication framework that enables the ground nodes to communicate with its peer ground nodes as well as the UAVs that act as relay nodes. The communication architecture is designed to work with existing MANET routing protocols.

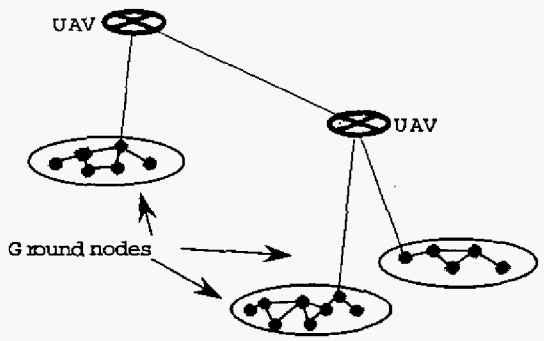

Fig. 1. MANET with three partitions and two connecting UAVs.

I. Introduction

Military ad-hoc networks are comprised of wireless nodes that are dispersed over a wide region and whose motion is governed by the tasks assigned to the nodes. The connectivity amongst the nodes depends on a number of factors. The transmission range of the nodes determines the distance based connectivity between the nodes. The nature of the terrain determines the propagation loss and therefore connectivity, thus two nodes which are within communication range may still not be able to communicate with each other due to the terrain induced path loss. Finally the mobility pattern is determined by the specific tasks that are assigned to them. It is then reasonable to expect that the network will not be fully connected at all times. Connectivity amongst all the nodes in the network is a desirable feature for military networks. Also, it might be important to ensure that certain high priority nodes in the network always remain connected. In this paper we address the issue of connecting several disconnected mobile ground sub-groups by dynamically placing aerial platforms such as UAVs to provide and maintain connectivity. It is obvious that the minimum requirement for full connectivity of the network is for each sub-group to have at least one node communicating with a UAV. Since the UAVs are scarce and expensive resources, the goal is to find the minimum number of required UAVs and their locations to have a fully connected network (Figure 1).

To our knowledge, the UAV placement in conjunction with the multi-hop routing capability of the ad-hoc networks is not addressed before. Similar works (e.g. [1], [2]) have mainly addressed other aspects of the UAV placement when enough number of UAVs are placed in the network to provide direct coverage for all nodes.

This material is based on work supported by the Space and Naval Warfare Systems Center - San Diego under Contract No. N66001-00-C-8063. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not neccessarily reflect the views of the Space and Naval Warfare Systems Center - San Diego.

We propose methods that lead to near-optimal solutions for both the number and locations of the UAVs for any given configuration of the ground nodes. In real applications, the position of the ground nodes and their direct connections with other nodes are updated in fixed time intervals and the algorithm is executed each time to give the updated locations (and the number) of the UAVs. The introduction of the aerial platforms results in a two-layer ad-hoc network and can be generalized to a multi-layer hierarchical network. We provide a communication framework by which a ground node is able to talk to its peer network as well as the network of aerial platforms. Similarly the UAV is able to communicate with its peer UAVs as well as the ground nodes within range. This framework allows us to use any of the existing on-demand MANET routing protocols. In our model we use DSR [3] as the routing protocol. Also, in order to simulate a realistic network scenario we use a modified version of 802.11 that extends the communication range up to $6Km$ [4].

Our simulations compare our algorithm with an idealized grid algorithm (exhaustive search) to determine optimality of the solution and desirable attributes of the algorithm in terms of the ground coverage. The communication architecture is validated using MATLAB and OPNET simulations.

The rest of the paper is organized as follows. In Section II we discuss the aerial vehicle placement algorithm in detail. Our formulation represenets the problem as an extension of the well-known Facility Location problem and our method provides a heuristic solution for that problem. In Section III we discuss the hierrachical architecture established by the UAV nodes and the routing that enables the ground nodes to communicate to the aerial vehicles. In Section IV we discuss the simulation environment in detail and present the results. Section V concludes the paper.

II. Aerial vehicle placement and tracking

The UAV placement algorithm takes as input the connectivity matrix of the nodes and the current location of the nodes. The connectivity matrix is generated based on distance and terrain constraints. The algorithm uses the node locations and their movement history to predict future locations. Clusters or partitions are detected using the connectivity matrix and this information is used by the algorithm to determine the minimum number of UAVs and their optimal locations for each time instant. The algorithm is thus called periodically to update the UAV paths based on the new locations of the nodes.

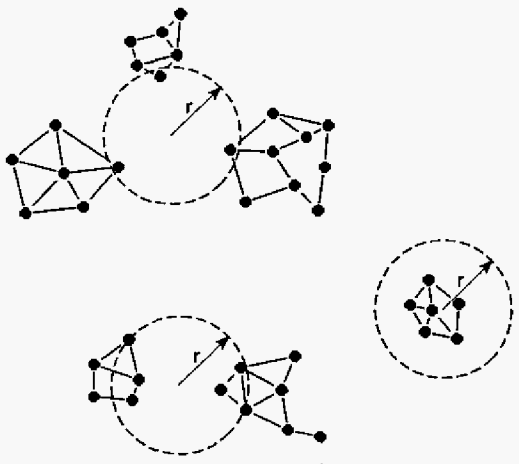

Fig. 2. Five clusters covered by three circles.

A. Problem Formulation

Let us denote by N the total number of ground nodes, M the number of subgraphs(clusters) in the network and by C_i; $i = 1, \ldots, M$ each of those clusters. By definition, each node in a cluster only communicates with the other nodes in the same cluster. We assume that all nodes have the same altitudes and the UAVs fly at a constant altitude h. The maximum node-to-UAV communication range is L which, together with h, defines the maximum coverage radius $MaxRadius = \sqrt{L^2 - h^2}$ on the ground for each UAV which is significantly greater than the maximum range for ground-ground communications. By definition, as long as at least one node from any cluster is within the communication range of a UAV, all nodes of that cluster can connect to the UAV via that node. The problem is therefore to find the minimum number of circles with radius $MaxRadius$ and their centers in such a way that at least one node from each cluster is within one circle. Figure 2 presents a graphical representation of our definitions.

Finding the exact solution of this problem involves exhaustive search on the different ways the nodes can be selected from each cluster and the ways clusters can be grouped together for coverage by single UAVs. It is not difficult to show that the computational complexity of this search is non-polynomial particularly when we consider it as an extended version of the Facility Location Problem [5], [6], [7]. In the Facility Location Problem, the goal is to find the locations and the number of facility centers, characterized by their radius of coverage, such that *any* point is covered by at least one facility. Our problem involves another degree of complexity for choosing the *best* nodes from each cluster.

In the following, a mathematical formulation of what

we call the Single-UAV Problem is presented and a number of its properties are discussed. We will show that this problem is in general a non-convex minimization problem with possibly multiple local minima. We then introduce a heuristic algorithm for the Single-UAV problem and use it to construct an algorithm for the Multiple-UAV Problem.

B. The Single-UAV Problem

In the Single-UAV Problem the objective is to find the location of the smallest circle that contains at least one node from each cluster. Notice that here we do not impose any limit on the radius of the circle. We denote by $N_i; i = 1, \ldots, M$ the number of nodes in each cluster and by $x_j^i; i = 1, \ldots, M; j = 1, \ldots, N_i$ the locations of the N_i nodes of cluster i. Using the above definitions, the problem of finding the smallest circle is in fact the minimization problem:

$$\min_{X \in R^2} \max_{i \in \{1,\ldots,M\}} \min_j \|X - x_j^i\| \quad (1)$$

In other words, the maximum distance between the center of the circle and the clusters should be minimized. The resulting X will determine the center of the circle and its radius is the value of the $\max_{i \in \{1,\ldots,M\}} \min_j \|X - x_j^i\|$ function at X.
This function is in general a non-convex function of X and therefore the problem cannot be solved by standard convex optimization methods. In fact, it is not difficult to find example cases where this function has multiple local minima. It should be mentioned at this point that for the case with only two clusters, the problem reduces to finding the closest pair of nodes, each belonging to one of the clusters, and placing the center of the circle in the middle of the line connecting the two such nodes. We are therefore more interested in finding an algorithm for covering three or more clusters.

Our typical problem settings are military applications where the clusters are groups of vehicles or soldiers moving in formations. In such cases, members of each group form connected graphs but, the graphs of different groups may or may not be connected to each other depending on the distance between the groups. our algorithm is based on the implicit assumption that all clusters more or less have smooth and convex shapes although the absence of this condition will not void the algorithm and as we will show in the results, it performs quite well in scenarios where no such restrictions are imposed on the clusters.
The most important part of the algorithm is to find the best nodes from each cluster that fit altogether in a circle with minimum radius. This fact can be observed in the three clusters in the upper part of Figure 2 where

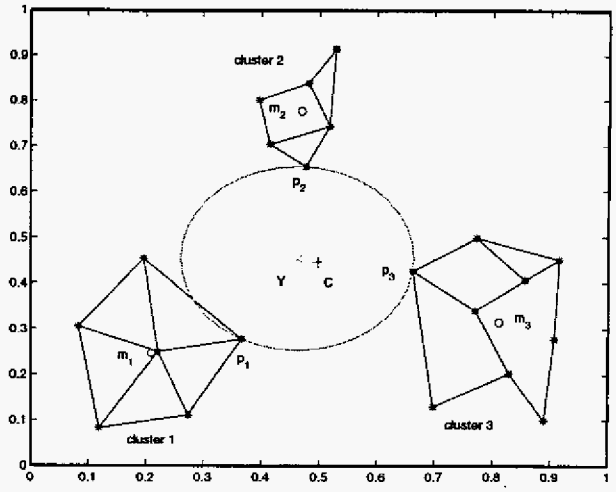

Fig. 3. Graphic representation of the single-UAV algorithm.

any other choice for the nodes from each of these clusters would result into a larger covering circle. Once the candidate nodes from the clusters are chosen, it is only needed to find the minimum covering circle for those nodes. The Minimum Covering Circle for a number of points is a well-known problem with both analytical and geometrical solutions having been proposed for it[8], [9], [10]. Our Single-UAV algorithm uses a virtual center point C to find the closest node p_i from each cluster to that and uses this set of nodes to find the minimum covering circle. The point C is the mean point or center of gravity(CG) of all clusters. However, to prevent the clusters with large number of nodes to have a greater effect on the location of C, it is calculated as the CG of the mean points m_i of all clusters. A sample case using this method is shown in figure 3 which results into location Y for the UAV. Obviously, for the case with three clusters, all three p_i nodes will fall on the perimeter of the covering circle (except when the three points lie on a straight line). The sequence of steps for this algorithm is summarized below

1) Calculate the center points $m_i; \ i = 1, ..., M$ for all clusters
2) Calculate the center C of all m_i points
3) Find from each cluster i, node p_i which is closest to C
4) Find the UAV location by solving the convex constrained minimization problem

$$\min_X R \quad s.t. \ ||X - p_i|| \leq R; \ i = 1, \ldots, M.$$

It should be pointed out that this algorithm does not take into account the coverage range of the UAV and simply tries to find the smallest circle covering at least one node

of every cluster. We will use this algorithm in the next section to build our UAV placement algorithm with the range constraints.

C. The UAV placement algorithm

Having explained a method for the placement of a single UAV, the main function of the general algorithm is to group the clusters into what we call superclusters in such a way that each supercluster can be covered by a single UAV and the number of superclusters is minimized. Simulation results comparing the performance of our algorithm with an exhaustive search method for a large number of node formations show that our algorithm performs very close to optimal. The UAV placement algorithm is explained in the following.

Algorithm 1 shows the sequence of computations and decisions of our algorithm. This algorithm is executed periodically at time steps $\ldots, t-\Delta t, t, t+\Delta t, \ldots$ and at any time t it calculates the number and locations of the UAVs for the next time step based on the prediction of the node locations for $t+\Delta t$. The locations of the nodes at time $t+\Delta t$ are calculated by a linear estimation based on the locations at $t-\Delta t$ and t. The connectivity matrix is then constructed based on the new locations. Disconnected clusters and the nodes belonging to each of them are detected using the connectivity matrix. In order to find a proper grouping of the clusters, we calculate the *distance* between any two clusters which is defined as the smallest distance between all points in one cluster to all points in the other cluster. It is obvious that if two clusters are more than $2*MaxRadius$ apart they cannot be covered by a single UAV and therefore they should not be placed in the same supercluster. The neighborhood relation between two clusters is defined as a 0 or 1 relation such that two clusters are considered neighbors if and only if their distance is less than $2*MaxRadius$. After calculating the inter-cluster distances, the neighbors of each cluster are easily found. With this definition, only neighbor clusters can be potential candidates for the same supercluster. Based on this observation, the algorithm starts with the cluster that has the smallest number of neighbors and chooses those neighbors which are also each other's neighbors. This potential set of clusters for coverage by a single UAV is then passed to the Single-UAV algorithm defined above to calculate the location and radius of the UAV that covers them. After finding the p_i points of the clusters and calculating the minimum covering circle for them, if the radius turns out to be greater than $MaxRadius$, the cluster with its p_i point farthest from the center is released and the algorithm is repeated for other remaining neighbors and the original cluster itself until a circle with a radius smaller than $MaxRadius$ is achieved. At this time, the covered clusters are deleted from the list and the algorithm is repeated for the next cluster with smallest number of neighbors and continues until all clusters are covered.

Once the number and locations of the required UAVs for time $t+\Delta t$ are computed, the algorithm decides which of the current UAVs (at time t) and possibly new UAVs should be dispatched to each of the new locations. This is done by comparing the current $k_i(t)$ and computed $k_i(t+\Delta t)$ locations of the UAVs and finding the matching that minimizes $0.5*mean(d)+0.5*var(d)$ where d is the vector of distances that the UAVs should move to reach to their assigned locations. This expression tries to minimize the total travel distance of the UAVs while keeping the distance traveled by all UAVs as same as possible. A similar approach is used for the cases where more or lesser number of UAVs are needed in the next time step.

III. Routing Architecture

As described in Section I the presence of the UAVs results in a two-layer adhoc network, i.e., the ground nodes are able to communicate amongst themselves either over the ground adhoc network or via the UAVs. Clearly, if partitions exist the only available routes are those provided by the UAVs. Note that a packet sent by a ground node might have to travel multihop over the ground network before it can reach an UAV. Each ground node has two interfaces, a primary (ground) interface to communicate with other ground nodes and the backbone (UAV) interface to communicate with the UAVs. Thus when a ground node tries to discover new paths it will send out the route discovery packets over the ground or the primary interface as well as the backbone or the UAV interface resulting in the discovery of separate multiple paths over the ground network as well as the aerial network. Another benefit of having separate interfaces is to provide different bandwidth capabilities on each interface allowing for QOS provisioning and load balancing in the network.

In order to remain independent of the ground network routing protocol we consider our aerial vehicles to provide the same routing functionality as any ground node except that the UAV will have an extended coverage. This allows us to utilize the existing routing protocols developed for MANETs and more importantly avoids the need for an auxillary protocol that coordinates the arrival and the removal of the UAVs with the routing protocol. Typical MANET protocols are either classified as reactive or proactive, where, in the first case routes are repaired/managed as reactions to route failures and in the latter case routes to destinations are established/maintained on a periodic basis. In our framework

Data : Δt: time step
$MaxRadius$: coverage radius of each UAV
N: number of nodes
$X_i(t)$; $i = 1, \ldots, N$: current locations
$X_i(t - \Delta t)$; $i = 1, \ldots, N$: previous locations
Result : U: number of UAVs,
Y_u; $u = 1, \ldots, U$: UAV locations
begin
- Predict node locations at $t + \Delta t$:
$X_i(t + \Delta t) = 2X_i(t) - X_i(t - \Delta t)$
- Calculate the $N \times N$ connectivity matrix
- Find the number of clusters M and their nodes
if $M > 1$ **then**
- Calculate the $N \times N$ matrix D of the node distances
- Calculate the $M \times M$ matrix E of cluster distances
- Calculate the $M \times M$ matrix P of cluster neighbors
if $E(i, j) < 2 * MaxRadius$ **then**
$P(i, j) = 1$
else
$P(i, j) = 0$
end
- Rearrange rows and columns of P to form all-1s diagonal blocks
- Set $u = 0$
while P *is not nil* **do**
- Set $u = u + 1$
- Find cluster c with the smallest number of neighbors
- Pick the largest square block B of matrix P that contains cluster c and name its clusters as $c_1, \ldots, c_b$
- Set $R = \infty$
- Find the center points m_i of all clusters in block B
repeat
- Find the overall center point C of the m_i; $i = 1, \ldots, b$ points
- Find, from each cluster c_i, node p_i which is closest to C
- Find the center Y_u and radius R of the smallest covering circle for nodes p_i; $i = 1, \ldots, b$
if $R > MaxRadius$ **then**
- Drop cluster i; $i \in \{1, ..., b\}$, other than c, with the largest distance from p_i to C, from block B
- Set $b = b - 1$
end
until $R \leq MaxRadius$
- Store Y_u
- Remove the remaining clusters in B from P
end
end
end

Algorithm 1: Finding the number and locations of the

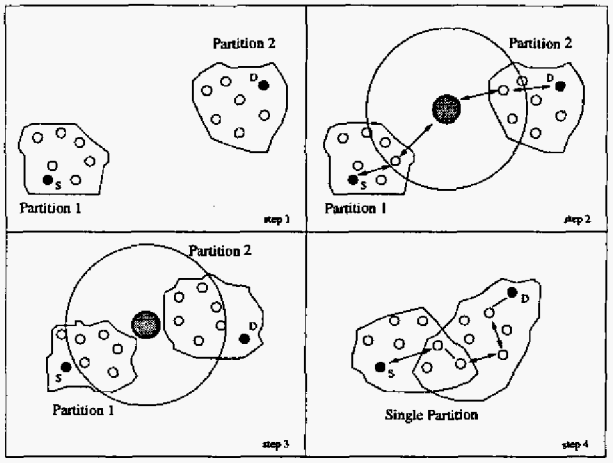

Fig. 4. Routing amidst partitions and merging

we consider reactive protocols as the latency resulting from route changes is lower in such protocols. Routing protocols are also classified as "on-demand" where in a node that had packets to be sent to a destination will continously look for a path to that destination till it is established and will reinitiate the search once the established route is lost. Such protocols are ideal for our routing framework as it supports all possbile cases where we might use an UAV to provide connectivity. Moving an UAV to certain location is equivalent to introducing a new connecting node, so it is imperative that the routing protocol continuously look for a path which previously could not be established due to partitions. This results in the formation of a route with the UAV as the connecting node (albeit multihop). The "on-demand" nature also helps in network recovery when UAVs are withdrawn from certain locations due to the merging of ground partitions. In this case, a ground node communicating via an UAV suddenly finds the route broken as the UAV has been removed, this prompts the routing protocol to look for alternate routes and a new route is established via the ground network which has merged and is the reason for removal of the UAV. Therefore any reactive on-demand protocol can provide routing in our architecture.

In our framework we have considered AODV [11] and DSR both of which are popular reactive on-demand protocols. In this paper, we only present the scenarios with DSR. When a node has packets to send to a destination, the node generates a route request packet and broadcasts it over the ground interface as well as the UAV interface. Intermediate nodes (UAV or ground) retransmit the packet until it reaches the destination. The destination responds with a route reply which backtracks the path traversed by the route request. In our DSR model route replies are sent for every route request that

is received. This allows the source to choose the path it wants to use. Currently the source chooses the first path that is established. We are working on an improved scheme to choose paths based on QOS metrics providing the capability to control the access to the aerial network and to perform traffic or entity based QOS provisioning. Figure 4 describes the various aspects of the routing. Source 'S' and destination 'D' are in different partitions (step-1). The source continuously searches for paths by generating route requests. Upon arrival of the UAV in (step-2), the partitions are merged and the the route request propagated by the UAV reaches the destination and therefore a path is set up and data transfer ensues. As time progresses the partitions merge (step-4) at which point the UAV is removed. This results in a Route failure and the source again sends route request messages. This time the route request propagates to the destination via the ground network and data transfer resumes. Thus the routing protocol adapts to the current state of the network. In Section IV we will discuss the impact of the UAV network on the performance of the routing protocol in terms of the routing overhead and protocol performance.

The above discussed approach totally neglects the advantages of the aerial network. Clearly, there can be routing approaches that can be developed that utilize the benefits of the aerial vehicles to improve routing overhead and network performance. However, this approach would require better coordination between the traffic requirements in the network and the UAV placement algorithm. We are currently looking at approaches wherein the UAV nodes are advantaged nodes and perform some form of intelligent routing while still being compatible with the existing MANET routing protocols.

IV. Simulation environment and results

A combination of OPNET 10.0.A [12] and MATLAB was used to evaluate the performance of our UAV algorithm and the routing architecture. MATLAB is used to evaluate our algorithm with respect to the grid algorithm and determine other performance metrics of our algorithm. The routing framework and the network model is evaluated in OPNET. The UAV placement algorithm is run in MATLAB and is interfaced with OPNET to interact with the routing framework.

In our studies we consider a network area of $10km \times 10km$. The network consists of $100 - 200$ nodes placed randomly in the form of 10 groups. We consider a reference point based group mobility model for the nodes. For each group, the reference point is altered based on the random direction model and is advanced in a random direction by a fixed distance ($20m$). Each node of the group is first displaced by the the same fixed

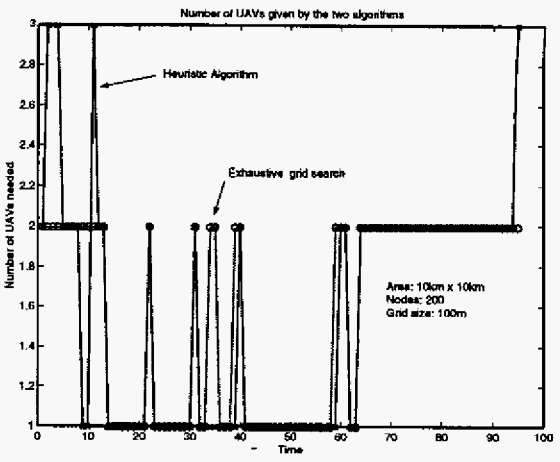

Fig. 5. Comparison of the number of UAVs given by the Grid and our heuristic algorithm

distance and then a perturbation is added whereby the node moves randomly to some location within a square of side $10m$ within the group confines. (this ensures that the nodes are traveling in their respective groups at speeds distributed between $10 \rightarrow 30m/s$).

The first set of results deals with the main objective of our algorithm i.e. the required number of UAVs to provide full connectivity for the network. We run the mobility model for 100 time steps to obtain 100 different formation of the nodes and compare the performance of our algorithm with the performance of an exhaustive search method that gives near-optimal results. The exhaustive search is performed by setting up a grid network with 100 meters spacing in both x and y directions. For each formation of the nodes, all possible options for placing 1 and higher number of UAVs in the grid points are examined and the first configuration with the smallest number of UAVs that provides full coverage for all nodes is selected as the optimal result. Although the discretization of the area limits our search to a finite number of points, the 100 meters spacing provides a close approximation to the optimal result. Figure 5 shows the results of both algorithms. As can be seen in the graph, the results of the two algorithms are equal in most cases with each algorithm over-performing the other at some points. Aside from the networking applications, the results indicate that our algorithm provides a low-complexity heuristic for the extended facility location problem defined above.

One of the important factors from a networking point of view is the number of times the nodes have to switch from one UAV to another. This measure has direct consequences in the number of routing messages generated in the network. Figure 6 shows the histogram of the number of nodes that change their UAVs in two

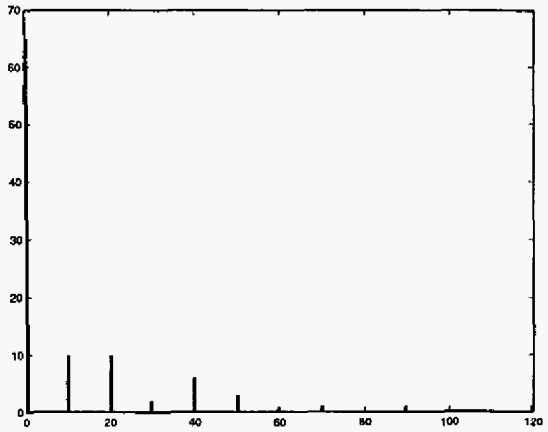

Fig. 6. Histogram of the number of nodes that change their associated UAVs

consecutive time steps. This histogram has an average of 10.5 nodes per step. In other words, on average only about 7 percent of the nodes change their UAVs in any time step.

We used OPNET to run the second set of experiments that deals with the performance of the algorithm in conjunction with a two-layer mobile ad-hoc network where each node has the complete protocol stack running DSR as the routing protocol over 802.11 as the MAC layer protocol. The communication medium is broadcast and nodes have bi-directional connectivity. The propagation model is the free space distance model . The ground nodes have a communication range of $1km$ between them. The UAV to ground node range is $3km$ and the inter-UAV communication range is $6km$. The UAV can fly between altitudes of $2k - 3km$. In our network, 4 sources are randomly chosen and these send data packets to random destinations. The application data is fixed length ($1024bits$) packets generated at uniformly distributed ($0 - 1sec$) inter-arrival times. We ensure that the source and the destination belong to different groups.

The results show the performance of the network with and without the UAVs and the placement algorithm. The top graph in Figure 7 shows the amount of traffic sent out by the source. The second graph shows the packets received by the destination node when the UAVs are not present. Clearly since most of the time the network is partitioned a large number of the packets are dropped. The last graph shows the packets received by the destination in the presence of the UAVs. As the UAVs provide full connectivity the destination receives most of the packets. This graph clearly shows the benefit of using UAV in the network to provide and improve connectivity.

Figure 8 shows the number of routing packets generated in the both the scenarios. In the scenario without UAVs the routing protocol (DSR) continuously sends

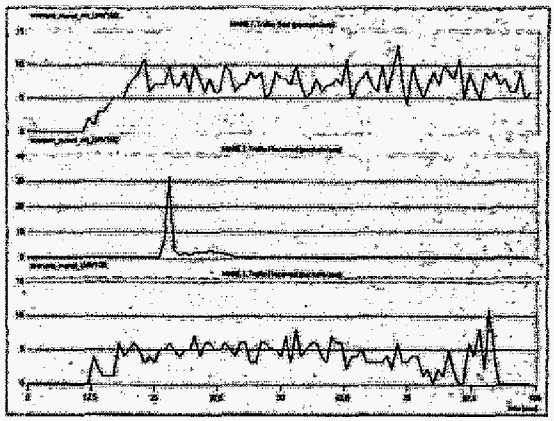

Fig. 7. Traffic sent and received for the scenario

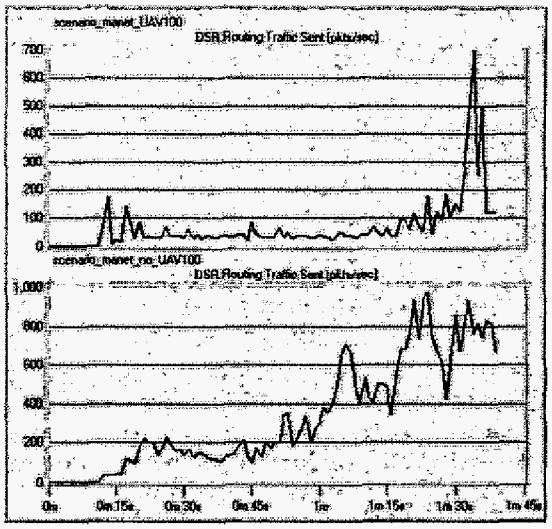

Fig. 8. DSR Routing packets generated

route requests to look for routes even if the destination is unreachable due to partitions. As can be seen from the second graph of Figure 8 this overhead can be significant and more importantly inconsequent as a route cannot be established. The first graph shows the routing overhead when UAVs are placed to improve connectivity. Clearly the overhead is much lower, as once the routes are established, routing messages are generated only for route repairs. The few spikes that occur indicate route failures probably due to changes in allocated UAVs resulting in network-wide route discovery.

Figure 9 shows the cumulative distribution of the route discovery time for the two cases. It is clear that the presence of the UAVs significantly reduces the route discovery latency. This is so as the UAV network can not only patch partitions and thereby provide connectivity but also improve the hop connectivity of existing ground routes.

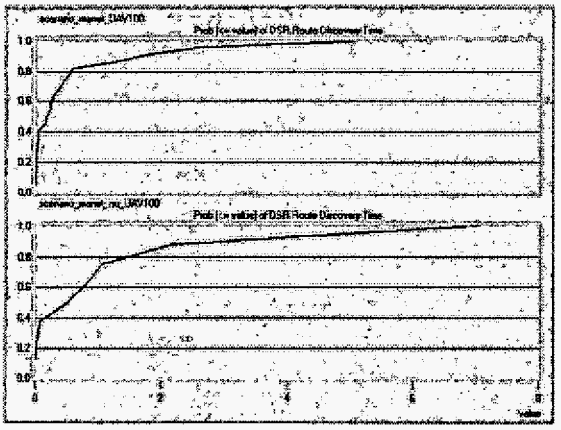

Fig. 9. CDF of route discovery latency

The results are preliminary and validate the performance of the UAV algorithm as well as the communication framework. Further detailed studies analyzing the routing overhead and the hop distribution are in progress and will appear in the final version of the paper.

V. Conclusions

In this paper we visited the problem of providing connectivity for mobile ad-hoc networks using Unmanned Aerial Vehicles (UAVs). Unlike the traditional approaches that provide more number of UAVs to cover all nodes, Our particular attention was on finding the minimum number of UAVs taking into account the multi-hop routing capability of the ad-hoc networks. We showed that this problem is an extension of the Facility Location problem and is a non-convex minimization problem. We also provided a heuristic algorithm for solving the problem and compared its results with an exhaustive search algorithm. We also provide a communication framework that enables the ground nodes to interact with the UAVs using existing MANET protocols.The results showed that the algorithm provides promising results for a set of nodes with group mobility movement. We also evaluated the performance of the algorithm, using OPNET simulations, in a mobile ad-hoc network with nodes running the DSR as the routing and 802.11 as the MAC layer protocols. The results in their current state show the effectiveness of the algorithm in conjunction with these protocols. More detailed experiments to evaluate the performance of the algorithm are in progress.

References

[1] Izhak Rubin and Runhe Zhang. Performance behaviour of unmanned vehicle aided mobile backbone based wireless ad hoc networks.

[2] S. Benjamin and I.Rubin. Connected disc covering and applications to mobile gateway placement in ad hoc networks. In *Proceedings of ADHOC NetwOrks and Wireless (ADHOC-NOW).*

[3] D.B. Johnson and D.A. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing, Kluwer Academic Publishers*, 1996.

[4] Kin K. Leung, Bruce McNair, Jr Leonard J. Cimini, and JackH. Winters. Outdoor ieee 802.11 cellular networks: Mac protocol design and performance. 2002.

[5] D. Ghosh. Neighborhood search heuristics for the uncapacitated facility location problem. *European Journal of Operational Research*, vol.150, no.1, 1 Oct. 2003. p. 150-62.

[6] A. Meyerson S. Guha and K. Munagala. A constant factor approximation algorithm for the fault-tolerant facility location problem. *Journal of Algorithms*, vol.48, no.2, Sept. 2003. p. 429-40.

[7] S. Guha and S. Khuller. Greedy strikes back: improved facility location algorithms. *Journal of Algorithms*, vol.31, no.1, April 1999. p. 228-48.

[8] D.J. Elzinga and D.W. Hearn. The minimum covering sphere problem. *Management Science*, 19(1), 1972, pp. 96-104.

[9] T.H. Hopp and C.P. Reeve. An algorithm for computing the minimum covering sphere in any dimension. *Technical Report, National Institute of Standards and Technology (NIST)*, NISTIR 5831, May 1996.

[10] N. Megiddo. The weighted euclidean 1-center problem. *Mathematics of Operations Research*, Vol. 8, No. 4, Nov. 1983, pp.498-504.

[11] C. E. Perkins and E. M. Royer. *Ad hoc On-demand Distance Vector (AODV) Routing*. Internet Draft, draft-ietf-manet-aodv-02.txt, November 1998. Work in progress.

[12] Opnet modeler version 10.0.a. www.opnet.com.