A Hybrid Ant-Colony Routing Algorithm for Mobile Ad-Hoc Networks

Shahab Kamali[1] and Jaroslav Opatrny[2]

[1] Unversity of Waterloo, Waterloo, Canada
skamali@cs.uwaterloo.ca
[2] Concordia University, Montreal, Canada
opatrny@cs.concordia.ca

Abstract. The dynamic nature of mobile ad hoc networks makes it difficult to consider a specific model for their topology which might change in a short period of time. Using the knowledge about the location of nodes, several relatively efficient position based routing algorithms have been proposed but almost all of them are sensitive to the network topology. Ant colony optimization based routing algorithms form another family of routing algorithms that usually converge to optimum routes. In our previous work we proposed POSANT, a position based ant colony routing algorithm for mobile ad-hoc networks. Although POSANT outperforms other routing algorithms in most cases, there are network topologies in which POSANT does not perform well. In this paper we introduce HybNet, a hybrid ant colony optimization based routing algorithm for mobile ad hoc networks which adapts itself to different network topologies. We carry out an empirical analysis of the performance of our algorithm and compare it with other routing algorithms. Our results show that HybNet almost always performs efficiently, even in some complex and variable network topologies.

1 Introduction

A mobile ad-hoc network might have a highly dynamic topology because mobile nodes can freely join it, leave it or move inside it. For example, a network may have more nodes in daytime than at night. Also the network graph might be dense in some parts and sparse in other parts. This dynamic nature of mobile ad-hoc networks increases the difficulty of message routing.

A *position based* routing algorithm uses the knowledge about the location of a node, its neighbors and the destination node to make a local routing decision at that node. To obtain the position of the destination node, position based routing protocols assume a location service (e.g. Greed Location Service(GLS)[11,3], Simple Location Service(SLS)[3] and Reactive Location Service(RLS)[3]) is available that provides location information on the nodes in the network. The strategy adopted by a location service is out of the scope of this paper and we just assume that a location service provides such information.

GPSR[9], *Compass routing*[10] and *Greedy routing* [6] are examples of reactive position based routing algorithms. These algorithms perform very well in

J. Zhou (Ed.): Complex 2009, Part II, LNICST 5, pp. 1337–1354, 2009.

networks with specific topology characteristics but their performance might be poor in some other networks (the performance metrics are discussed later in the paper). Position based routing algorithms typically perform very well in dense networks, while in sparse networks they might fail to find a route or the found route might be much longer than the shortest path.

Algorithms which are based on *ant colony optimization* (ACO) form another family of routing algorithms. *ANTNET*[5] and *ANTHOCNET*[4] are examples of ant colony based routing algorithms. These algorithms eventually converge to routes whose lengths are very close to the length of the shortest path [5], [7]. In our previous work we proposed *POSANT*[8], a position based ant colony routing algorithm which outperforms the other ACO routing algorithms in terms of convergence time or routing overhead in most cases. However, in some network topologies the performance of POSANT is relatively poor and its convergence time is much longer than other ant colony routing algorithm. To address this problem, in this paper we present a routing algorithm that, like POSANT, adopts a position based ant colony strategy. Unlike POSANT, which is based on some assumptions about the network topology (POSANT assumes the optimum route passes through nodes whose directions are close to the direction of the destination node), the strategy presented in this paper does not consider any specific assumption about the network topology. The routing algorithm presented in this paper is a hybrid routing algorithm that uses a combination of position-aware and position-unaware ant-colony strategies; so it is called HybNet. This algorithm performs well in networks with complex topologies where the other routing algorithms might face difficulties and perform poorly. The applications of this algorithm includes cases where the network topology includes dense and sparse regions, empty regions, and especial shapes and also cases where the transmission range of nodes can be highly irregular. In addition, HybNet is suitable for cases where a large amount of data is transmitted after route establishment and thus it is important to find optimum routes (i.e. an example is video and audio streaming).

In the next section we first describe the network model and give a definition of the routing problem. We then explain the related works including typical position based and ant colony based routing algorithms. Section 3 defines HybNet and Section 4 contains the simulation results of HybNet and a comparison with POSANT, ANTNET, ANTHOCNET and GPSR routing algorithms. Section 5 contains conclusions.

2 Network Model and Related Routing Algorithms

We represent a mobile ad-hoc network as a planar graph with an edge between each pair of nodes that can communicate directly (i.e. all communications are bi-directional and a node will maintain a link to another node if they are able to exchange messages directly). The transmission range of a node in a specific direction is a line segment in that direction from that point to the farthest point

it can directly send messages to. We assume each node can have different transmission ranges in different directions, as in the reality where the existence of an obstacle or noise makes the transmission radius of a node irregular. Furthermore due to different power consumption, different nodes may have different transmission ranges. Different regions of a network graph may have different topological characteristics, e.g. a network graph may be dense in some regions and sparse in other regions. It is assumed that each node knows its position, the position of its neighbors and the position of the destination node. To reach a destination, a message follows a path of intermediate nodes. The length of this path is measured in terms of hop-count (i.e. the number of edges on the path). The time required for a packet to move from one node to a neighbor node is assumed to be the same for all nodes (i.e. we do not consider distance between two neighboring nodes, buffering, congestion and other causes of packet delay in this paper). So the delay of a packet (i.e. the elapsed time since the moment that the packet is launched from the source node until the moment that the packet is received by the destination node) is a linear function of the length of its path to the destination. We suppose the route finding algorithm starts as soon as a data packet needs to be sent from a source to a given destination (reactive routing) so the routes should be established in the shortest possible time.

In the remaining of this section we briefly describe some related works.

GPSR[9] and *Compass routing* [10] are examples of position based routing algorithms. These algorithms do not guarantee to find a shortest path to the destination [6,2,9]. An example of a case when these algorithms find a path to a destination which is longer than the shortest path is presented in Figure 1(a).

Figure 1(b) shows an example where these algorithms fail to find a route to a destination.

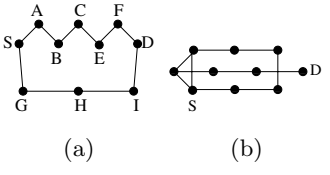

Fig. 1. (a) The route (from S to D) found by GPSR and DIR is SABCEFD while the shortest path is SGHID. (b) DIR and GPSR fail to find a route from S to D.

There are several other position based routing algorithms but they have similar shortcomings of either not guaranteeing to find a path to the destination or finding a path which is much longer than the shortest path [6,2]. Although position based routing algorithms have the above disadvantages, they have some useful characteristics: typically no routing table or passed traffic history is maintained and no control packet needs to be exchanged. So these algorithms are simple to implement and the overhead of routing is small. Moreover failures occur rarely when the network graph is dense [6].

Ant colony optimization (ACO) is a stochastic approach for solving combinatorial optimization problems like routing in computer networks. ANTNET [5], ANTHOCNET [4], ARA [7] and POSANT [8] are examples of ant colony optimization based routing algorithms. POSANT is an ant colony optimization based routing algorithm which uses the knowledge about the position of nodes as a heuristic to increase its efficiency. In general, ACO routing algorithms eventually converge to a group of routes which are very close in length to the optimum route [5,7,12]. In most cases the convergence of ANTNET is slow. ANTHOCNET and ARA broadcast an ant in the route discovery phase. Although an ant is a small control packet, broadcasting implies a large overhead on the network especially when the size of the network is large. In ANTHOCNET, an ant carries a list of its visited nodes, so its size will grow as it goes far from the source node and the routing overhead will increase. As a result, ARA and ANTHOCNET are not scalable. POSANT in most cases has a relatively short convergence time but in some cases its convergence time may be longer than the convergence time of ANTNET.

3 HybNet Routing Algorithm

In this section we introduce HybNet, a routing algorithm which performs efficiently in networks with complex topologies. While a complex topology or a fast topology transform might affect the performance of other routing algorithms, the efficiency of HybNet remains relatively high. HybNet has three main phases, route establishment, data transmission and link management. We suppose there is a set of data packets in a source node S, waiting for a route to be established between S and a destination. First, only forward and backward ants are transmitted and after the algorithm decides that the routes are established, data packets are transmitted.

Consider a destination node D and a network graph G. For each node N in G, we partition its neighbors into three zones called $zone_1$, $zone_2$ and $zone_3$. If H is one of the neighbors of N, θ_H is defined as the angle between line segments NH and ND. Node H belongs to $zone_1$ if $\theta_H \leq \pi/4$, $zone_2$ if $\pi/4 < \theta_H < 3\pi/4$, and $zone_3$ if $3\pi/4 \leq \theta_H \leq \pi$, see Figure 2 (this definition is similar to the definition of zones in POSANT [8]).

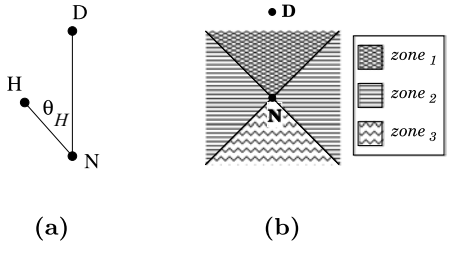

Fig. 2. (a) θ_H is the angle between NH and ND. (b) Different zones of N for destination node D.

3.1 Pheromone Trails

In HybNet, for a specific destination two pheromone trails, a *greedy trail* and a *regular trail*, may be assigned to each of the outgoing links of a node. For each node we assign these trails to each of its outgoing links if the node is reached by at least one forward ant heading to the destination. Suppose A is a node in the network that receives an ant for the first time heading to destination D, and B is one of the neighbors of A. To initialize the value of the greedy trail on $\boldsymbol{AB}$, we use a greedy policy, the assigned values decrease with the zone number. Having three values ν_{gr_1}, ν_{gr_2} and ν_{gr_3} such that $\nu_{gr_1} > \nu_{gr_2} \geq \nu_{gr_3}$, an amount of pheromone equal to ν_{gr_i} will be assigned to the greedy trail of $\boldsymbol{AB}$ if B belongs to $zone_i$. To the regular trail, that can also depend on the zone, one of the three values $\nu_{reg_1}, \nu_{reg_2}, \nu_{reg_3}$ is assigned. However, in our experiments we used $\nu_{reg_1} = \nu_{reg_2} = \nu_{reg_3}$.

Each node maintains a table which contains the values of the pheromone trails assigned to its outgoing links for different destinations. Whenever a forward ant for a specific destination arrives at a node, this table will be searched. If no pheromone trail for this destination exists, the pheromone initialization process assigns pheromone trails for this destination to all the outgoing links of that node. If a node doesn't receive any packet for the destination for a specific period of time, the corresponding pheromone trails will be deleted from the pheromone trail table of that node. This time period is defined to be in the order of seconds. Also the entries in the BR table (to be defined later in this section) for that destination will be deleted after that time.

3.2 Route Establishment

For establishing routes between a given source S and a destination D, HybNet launches n forward ants from S heading to D at regular time intervals. In our experiments we set n to be 1. Assigning big values to n might slightly decrease the route establishment time but it also increases the overhead of the algorithm. In addition to the pheromone trial table, each node maintains another table which we call *Back Routing (BR)* table. When a forward ant enters a node from one of its neighbors, an entry in the BR table will be created that stores the identifier of the neighbor the forward ant is coming from, the sequence number of the ant, and the identifier of the destination. At each node a forward ant makes a stochastic decision in two steps to select the next hop. In the first step it decides which of the greedy or regular pheromone trails to use for making a stochastic decision in the second step. In the second step, it selects the next hop stochastically using the values of the selected pheromone trails. In the destination node, each received forward ant will be dropped and a backward ant will be sent back to the source node. Using information in the BR tables, the backward ant, which includes the sequence number of the corresponding forward ant, takes the same route but in reverse to reach the source node. The decision of a forward ant to use either greedy pheromone trails or regular trails is marked in the BR table. When a backward ant reaches a node, the BR table will be searched for the

corresponding forward ant. Based on the type of the pheromone trail (i.e. greedy or regular) used by that forward ant, we call this backward ant greedy or regular backward ant (a regular backward ant in a node may be a greedy backward ant in another node). At each node, a backward ant reports the length of the traveled path from the destination to that node. Each node keeps the average and standard deviation of reported distances to the destination by the greedy and regular backward ants. For simplicity, we call them greedy average, greedy standard deviation, regular average and regular standard deviation. When a node receives a greedy or regular backward ant, it updates the greedy or regular standard deviations and averages.

To reduce the effect of old backward ants, we define two fixed size *window*s in each node that contain recently received backward ants. One window is assigned to greedy backward ants and the other one is assigned to regular backward ants. The average and standard deviation of reported distances to the destination will be calculated only for the backward ants residing in the window. When a new backward ant is received we put it in the corresponding window and discard the oldest ant if the maximum widow size has been reached. Selecting an appropriate maximum window size is important. If the maximum window size is too small, the average delay calculated from the window information would be too far from the real average. If the maximum window size is very big, existence of very old ants would affect the result for a long time. Suppose α_{gr}, σ_{gr} and α_{reg}, σ_{reg} are the average and standard deviation of the delays reported by greedy and regular backward ants residing in the corresponding windows. These averages are used by the future forward ants to decide on the use of greedy or regular trails for selecting the next hop. Suppose P_{gr} is the probability that greedy trails are used and P_{reg} denotes the probability that regular trails are used by a forward ant. P_{gr} and P_{reg} are calculated using Equation 1.

$$P_{gr} = \begin{cases} 1 & \text{if } \sigma_{gr} < t \text{ and } \alpha_{gr} < \alpha_{reg} \\ 0 & \text{if } \sigma_{reg} < t \text{ and } \alpha_{reg} < \alpha_{gr} \\ C_{gr} & \text{otherwise} \end{cases} \qquad (1)$$
$$P_{reg} = 1 - P_{gr}$$

In the above equation t is a threshold value and when the standard deviation of the reported distances to the destination is less than this value, we conclude they have taken paths with almost the same lengths. C_{gr} is a constant value that determines which type of pheromone trails has higher chance to be used at the beginning of the route establishment. In the second step, the ant should be forwarded to one of the neighbors using the values of the pheromone trails of the selected type. This is done using a stochastic decision similar to other ACO routing algorithms. Suppose the algorithm decides to use greedy trails to select the next hop. Consider a forward ant is currently residing in node N with k neighbors $H_1, H_2, ..., H_k$. Suppose $\phi_{gr\,i}$ is the value of the greedy pheromone trail assigned to $\boldsymbol{NH_i}$. The ant will select H_i as the next hop with probability p_i which is calculated using the following equation:

$$p_i = \frac{\phi_{gr\,i}}{\sum_{j=1}^{k} \phi_{gr\,j}} \qquad (2)$$

Algorithm 1. HybNet routing algorithm

```
{S is the source node, D is the destination node and C is the current node}
if C = S then
  put one compass packet in C's buffer
end if
for each clock time do
  if C = S then
    if one of the conditions in section 3.3 is true then
      put one data packet in C's buffer
    else
      put one forward ant in C's buffer
    end if
  end if
  for each message m in C's buffer do
    if (m→type = ForwardAnt) or
      (m→type = DataPacket) then
      NextHop = SelectNextHop()
      send m to NextHop
      if NextHop = D then
        m →type = BackwardAnt
      end if
    else if m→type = BackwardAnt then
      find NextHop in C's BackRouting table
      send m to NextHop
      IncreasePheromone(NextHop,m)
      update regular and greedy averages
      if NextHop = S then drop m
    else if m→type = Compass then
      find NextHop using compass routing algorithm
      send m to NextHop
      if NextHop = D then
        m →type = BackwardCompass
      end if
    else if m→type = BackwardCompass then
      find NextHop in C's BackRouting table
      send m to NextHop
      if NextHop = S then
        enter greedy mode and drop m
      end if
    end if
  end for
  Evaporate()
end for
```

An analogous equation is used if the algorithm decides to use the regular trails. Moving from node B to node A, a backward ant increases the amount of the corresponding pheromone trails stored in $\boldsymbol{AB}$. It uses a Formula 3 to update the greedy pheromone trail on $\boldsymbol{AB}$.

$$\phi_{gr\,\boldsymbol{AB}} = \phi_{gr\,\boldsymbol{AB}} + g(d) \times \omega(\boldsymbol{AB}) \tag{3}$$

In the above formula d is the number of traveled nodes from the destination to node B by the backward ant, $g(d)$ is a decreasing function of d, ω is a *weight function* and its value depends on the zone of A in which B is residing.

$$\omega(\boldsymbol{AB}) = \begin{cases} w_1 \geq 1 \text{ if B is in } zone_1 \text{ of } A \\ w_2 = 1 \text{ if B is in } zone_2 \text{ of } A \\ w_3 \leq 1 \text{ if B is in } zone_3 \text{ of } A \end{cases} \tag{4}$$

This weight function helps the algorithm to converge faster because in most cases the shortest path passes through nodes which are closer in direction to the destination.

For updating regular trails, we use Equation 5. In this equation g and d are the same as in Equation 3.

$$\phi_{reg\,\boldsymbol{AB}} = \phi_{reg\,\boldsymbol{AB}} + g(d) \tag{5}$$

An evaporation process modifies the value of pheromone trails in regular time intervals. It is done by multiplying the value of each pheromone trail by a number $\mu < 1$ at regular time intervals (Eq. 6). This is to reduce the effect of ants happened to take non-optimum routes to reach the destination as the time passes by.

$$\phi_{\boldsymbol{AB}} = \mu \cdot \phi_{\boldsymbol{AB}} \tag{6}$$

We complete the definition of HybNet by proposing an algorithm to enhance its performance. At the beginning of route establishment process, a special control packet which we call *compass packet* is sent to the destination. This packet uses *compass* routing algorithm [10] to reach the destination. Upon receiving this packet, the destination destroys it and sends a *backward compass* packet back to the source node. The backward compass packet takes the same path as the compass packet but in reverse to reach the source node. When the source node receives this packet, it enters the greedy mode. In this mode when launching a forward ant, the source node marks it by setting a flag. The first time a node like A receives a marked forward packet, it uses Equation 7 to update the greedy pheromone trails on its outgoing links and also increases the value of C_{gr} in Equation 1.

$$\phi_{gr\,\boldsymbol{AB}} = \phi_{gr\,\boldsymbol{AB}} \cdot u(\boldsymbol{AB}) \tag{7}$$

In the above formula B is one of the neighbors of A and $u(AB)$ is a weight function whose value is dependent on the zone of A in which B is residing.

$$u(\boldsymbol{AB}) = \begin{cases} u_1 > 1 & \text{if B is in } zone_1 \text{ of } A \\ u_2, \quad u_1 \geq u_2 \geq 1 & \text{if B is in } zone_2 \text{ of } A \\ u_3 = 1 & \text{if B is in } zone_3 \text{ of } A \end{cases} \tag{8}$$

If the marked forward ant is the first forward ant for a specific destination which arrives to A, the above update takes place after initializing the pheromone trails in the routing table of A.

The motivation of using a compass packet is that compass routing finds a route whose length is close to the length of the optimum route in most cases when the average degree of nodes is high. Moreover, the overhead of sending a single control packet in the network is negligible while using this packet may significantly reduce the convergence time. Here we considered compass routing algorithm because it is a robust porition based routing algorithm however, any other position based routing algorithm could be considered.

3.3 Sending Data Packets

Sending data packets starts after routes are established between the source and the destination. As we mentioned earlier, each node which is involved in the route establishment process maintains greedy and regular averages and standard deviations of the reported distances to the destination. To decide which time is the best to start sending data packets, HybNet uses the standard deviations and averages kept in the source node for that destination. Sending data packets too late increases delay and sending them too early increases packet loss and the delay caused by following bad routes. HybNet stops launching control packets from the source node and starts sending data packets instead, if one of the following conditions becomes true:

$$\begin{array}{l} \sigma_{gr_{src}} < t \text{ and } \alpha_{gr_{src}} < \alpha_{reg_{src}} \text{ or} \\ \sigma_{reg_{src}} < t \text{ and } \alpha_{reg_{src}} < \alpha_{gr_{src}} \text{ or} \\ \sigma_{gr_{src}} < t \text{ and } \sigma_{reg_{src}} < t. \end{array}$$

In the above, t is the threshold of Equation 1, and $\alpha_{gr_{src}}$, $\alpha_{reg_{src}}$, $\sigma_{gr_{src}}$ and $\sigma_{reg_{src}}$ stand for the average and standard deviation of the reported distances to the destination for greedy and regular trails of the source node. If $\sigma_{gr_{src}}$ is small enough, we can assume that at the source node the forward ants that decided to use the greedy trails in the first step (i.e. we mentioned that each stochastic decision is made in two steps) followed paths with almost the same length to reach the destination. The same can be assumed if $\sigma_{reg_{src}}$ is small enough. So when $\sigma_{gr_{src}} < t$ and $\sigma_{reg_{src}} < t$ we can say that the algorithm has converged to a route or a group of routes with similar lengths, and it is a reasonable time to start sending data packets. If the first condition is true then the ants which used regular trails at the source node in the first step experienced more delay than the ants used greedy trails. Also since $\sigma_{gr_{src}}$ is relatively small, the probability that this condition changes is low. Regarding Equation 1 for calculating the values of P_{gr} and P_{reg}, in this case the packets will use greedy trails to select the next hop and so will use the routes that the algorithm is converged to. The same could be used to justify the second condition. In HybNet, data packets are treated like forward control packets. Since ACO routing algorithms eventually converge to optimum or almost-optimum routes [5,12], at least one of the above conditions eventually becomes true and sending data packets will be started. Our experiments, presented in the next section, confirm that sending data packets starts relatively fast in practice.

3.4 Link Management

Failure recovery. If the link between two nodes like A and B breaks while a connection is running between a source S and a destination D, HybNet performs a failure recovery as follows. For each active connection, each involved node defines a mode that can have two values, *proper mode* and *broken mode*. Initially each node is in the proper mode. If A realizes that the link to B is broken and there are pheromone trials corresponding to link $\boldsymbol{AB}$ for D in the pheromone table of A, its mode will be changed to broken mode. In the broken mode, if A has another outgoing link to which pheromone trails are assigned for destination D, the stochastic data routing will be continued. Otherwise, A informs its neighbors that there is no route from A to the destination anymore. Upon receiving this message, each node enters broken mode and follows the same algorithm as if the link to A is broken. If A receives duplicate packets while it is in the broken mode, it will inform its neighbors that there is no route to D from A. It is to prevent loops after a link failure. When a node receives a backward ant which should pass through a broken link, the backward ant will be dropped. If the source node has only one outgoing link that contains pheromone trails for D and this link breaks or a message from this link is received that states there is no route to D, a new route establishment process will begin and sending data packets will be suspended until new routes are established. After a specific time which is defined to be in the order of milliseconds is passed from the moment that the link failure is detected, the mode of A will be changed to the proper mode.

Handling new links. When a new node joins the network, or when a node moves and approaches some other nodes, new links may appear in the network. Suppose node A figures out that it can directly communicate with a new node C. Also suppose A has routing information about some destinations like D in its pheromone table. HybNet assigns a greedy pheromone trail to link $\boldsymbol{AC}$ as follows. A calculates the average of the value of the greedy pheromone trails for destination D in its pheromone table. If C is located in $zone_1$ of A (i.e. for destination D), then a pheromone trail whose value is equal to this average will be assigned to $\boldsymbol{AC}$. Otherwise if C is located in the other two zones, the half of this average will be assigned to this link. Similarly we assign a regular trail for each destination to $\boldsymbol{AC}$. The only difference is that we assign the average of the value of regular pheromone trails regardless of the zone of A in which C is residing. In the case that node C does not have routing information for D (i.e. because node C is a new node or it is not reached in the routing process before), after it receives a packet heading to D, a pheromone initialization process (i.e. as explained before) assigns pheromone trails to its outgoing links. The same approach is applied in node C for assigning pheromone trails to $\boldsymbol{CA}$ (i.e. for the destinations that node C has routing information about them).

4 Performance Evaluation

In this section we evaluate the performance of HybNet and compare it with POSANT, ANTNET, ANTHOCNET and GPSR routing algorithms. To perform

Table 1. Parameters of different algorithms and their values in our experiments

Algorithm	Parameter	Value	Description
HybNet	n	1	number of generated forward ants at each clock time
	t	1	standard deviation threshold
	$(\nu_{gr_1}, \nu_{gr_2}, \nu_{gr_3})$	(20,1,1)	pheromone initialization
	$(\nu_{reg_1}, \nu_{reg_2}, \nu_{reg_3})$	(10,10,10)	pheromone initialization
	μ	0.95	pheromone evaporation
	$(\omega_1, \omega_2, \omega_3)$	(1.2,1,0.8)	used in Equation 4
	(u_1, u_2, u_3)	(10,1,1)	used in Equation 8
	c_{gr}	0.5	used in Equation 1
	window size	50	defined in route establishment phase
POSANT	t	1	standard deviation threshold [8]
	(ν_1, ν_2, ν_3)	(20,1,1)	pheromone initialization [8]
ANTHOCNET	α_1	2	defined in [4]
	a	20	defined in [4]

this evaluation, we simulated the mentioned routing algorithms and tried them on different sets of network graphs.

It is assumed that it takes one millisecond for each packet to go from one node to a neighbor node and this time is always the same all over the network. Table 1 lists the parameters of different routing algorithms considered in this section and their corresponding values used in our experiments.

4.1 Values of the Parameters

In this section, we evaluate the effect of each parameter of HybNet on its performance. To evaluate the effect of a parameter, we assign different values to it while the other parameters are fixed and try HybNet on a set of randomly generated network graphs. This set includes 100 graphs with 60 nodes distributed randomly and uniformly over an area of size 500×500. The transmission ranges of nodes vary from 50 to 70 units. For each network 10 source-destination pairs are selected randomly and the result is averaged. In the following comparisons, except the parameter whose effect is studied, the other parameters have the values listed in Table 1. The performance is evaluated in terms of *delivery rate* that is the ratio of the number of packets received by the destination node to the number of packets sent by the source node, and *average packet delay* that is the average time that takes for the sent packets to reach the destination. When all packets experience the same delay, we can conclude that the algorithm is converged to routes with almost the same length. So the standard deviation of delays can be used to estimate the average convergence time.

In Fig. 3 and 4, the effect of n, the number of generated forward ants at each clock time by the source node, on the performance of HybNet is studied. As these figures show, the performance becomes better when two forward ants are launched at each clock time, but it does not change significantly when n is 3.

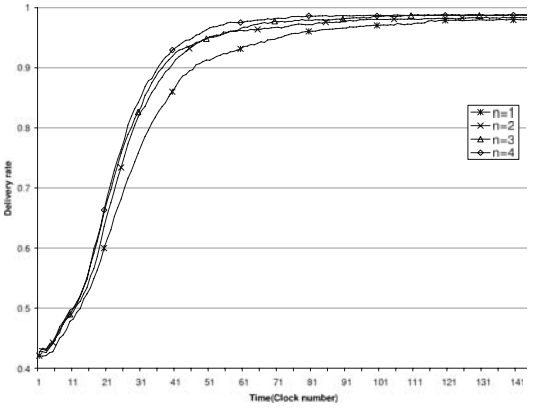

Fig. 3. Average delivery rate of HYBNET with different values assigned to n

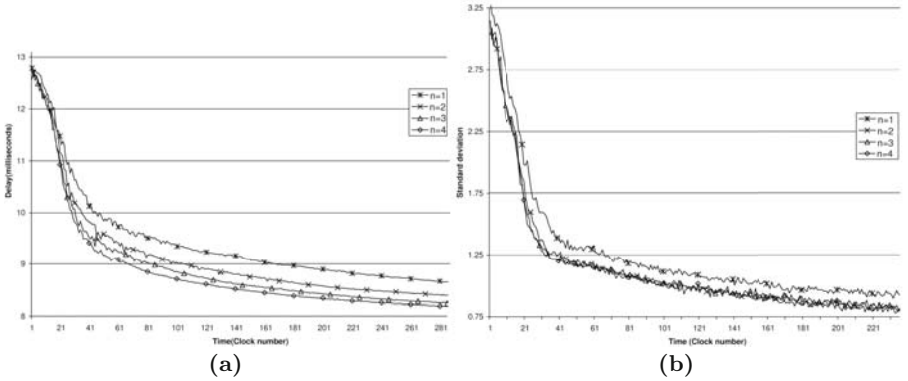

Fig. 4. (a) average (b) standard deviation of packet delay of HybNet with different values assigned to parameter n

The effect of the other parameters of HybNet on its performance is evaluated similarly but because of space limitations we do not present the results here.

4.2 General Network Topologies

In this section we compare the performance of different routing algorithms by trying them on a set of 200 randomly generated network graphs. Each graph has 90 nodes distributed randomly and uniformly over an area of size 500×500 square unit. The transmission range of each node varies from 40 to 60. For each network, 5 source-destination pairs are selected randomly and the results are averaged.

If a routing algorithm fails to deliver a packet to the destination, the sender must somehow detect this failure and try to resend the packet hoping that this

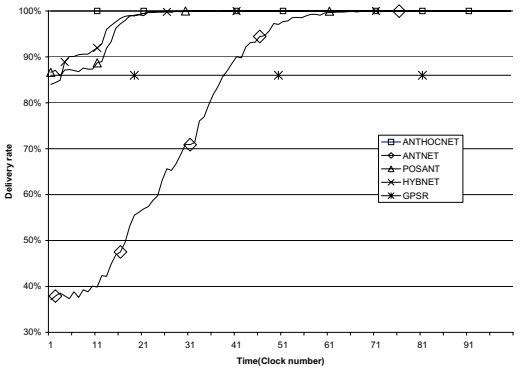

Fig. 5. Average delivery rate of HybNet, ANTNET, POSANT, GPSR, and ANTHOCNET

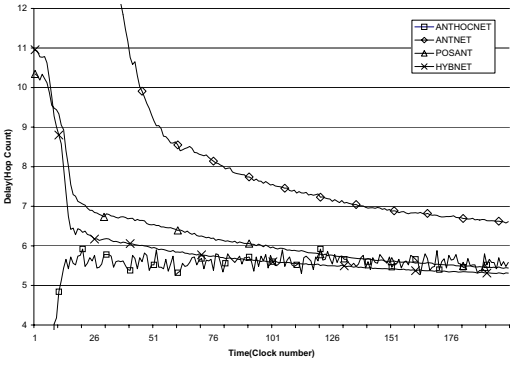

Fig. 6. Average packet delay of HybNet, ANTNET, POSANT and ANTHOCNET

time the algorithm delivers it. This increases the traffic in the network and also the delay experienced by the receiver. Thus it is very important to guarantee a high delivery rate. In Figure 5 the average delivery rate of HybNet, ANTNET, ANTHOCNET, POSANT and GPSR is compared. This figure shows how the average delivery rate varies with time. As the time progresses, the delivery rate increases and eventually becomes almost 100% for all algorithms except GPSR. As the graph shows, HybNet has the highest delivery rate among the others after ANTHOCNET (which broadcasts ants). GPSR has a relatively low delivery rate. It is because this algorithm fails in some cases as a result of irregular transmission ranges of the nodes. ANTNET reaches 100% delivery rate slower than the other ant based algorithms. The average packet delays are compared in Figure 6. This figure shows how average packet delays of the algorithms vary by time. The average delay of HybNet reaches to its minimum faster than the others. As the time passes on, all the algorithms converge to the paths with almost the same lengths and the packets experience almost the same delay. Because of the relatively low delivery rate of GPSR in this network model, this algorithm is not considered for this comparison. The reason for the fluctuations in ANTHOCNET

graph is that we chose α_1 to be 2 which means routes which are longer than the shortest path by at most 2 hops are also used for packet routing.

4.3 Complex Network Topologies

The effect of highly irregular network graph shapes on the performance of different routing algorithms is studied in this section. As we mentioned earlier, one of the main contributions of this paper is to propose a routing algorithm that unlike other routing algorithms, performs well regardless of the network topology characteristics. Although POSANT, which uses information about the position of nodes to enhance its performance as an ant colony routing algorithm, performs fairly good in most cases, there are special cases that make this algorithm to perform poorly. As an example consider the graph in Figure 8. To make a comparison, we generated 100 networks with nodes distributed over an area whose shape is shown in Figure 7. Each graph has 60 nodes. Each node is placed randomly within the area except the source and destination nodes which are placed in fixed points shown in Figure 7. This is a challenging example for most routing algorithms. In Figure 9 the average delivery rate of the different algorithms is compared. HybNet has the highest delivery rate after ANTHOCNET. GPSR has a very low delivery rate so it is not presented in this graph. POSANT has a relatively poor delivery rate at the beginning. Figure 10 compares the average delay of the mentioned routing algorithms. Because GPSR has a very low delivery rate, it is not considered for this comparison. HybNet has the shortest delay after ANTHOCNET. Also as the graph shows, ANTNET has a shorter convergence time than POSANT. This comparison shows that in some

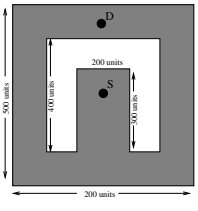

Fig. 7. Nodes of the graphs are distributed in the shaded area

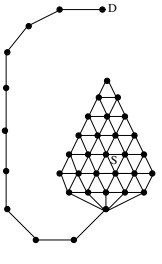

Fig. 8. A complicated network topology regarding the placement of source S and destination D

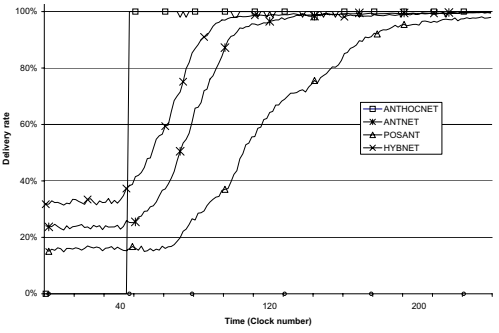

Fig. 9. Average delivery rate of different routing algorithms in a set of graphs with a special shape

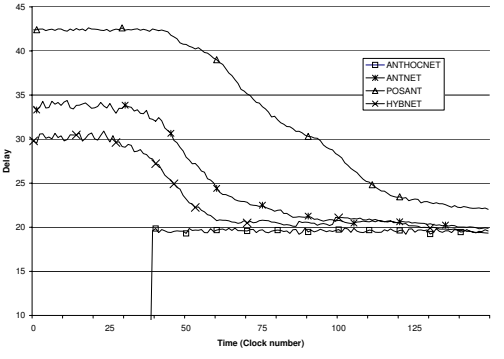

Fig. 10. Average delay of different routing algorithms in a set of graphs with a special shape when the source and destination nodes are fixed

network graphs where nodes are spread over a very irregular area, the convergence time of POSANT might be very long. Also in these cases position based routing algorithms may have a very low delivery rate. HybNet has a relatively short convergence time in these cases.

4.4 Mobility of Nodes

In this section the performance of HybNet when nodes move inside the network is evaluated. We used a variation of Manhattan mobility model [1] in which nodes move at a constant speed. In this mobility model, each node takes a step in the same direction as its previous step with probability 50%. Also it can take an orthogonal step (i.e. turn left or right) with probability 25% (i.e. for each direction). The results presented in this part are acquired after trying HybNet on a graph with 100 nodes spread over an area of size 500×500. The transmission range of nodes is between 50 and 70. In each experiment each node starts from the same initial position and moves regarding the above movement pattern independent of the other nodes. In Figure 11, the average delay and

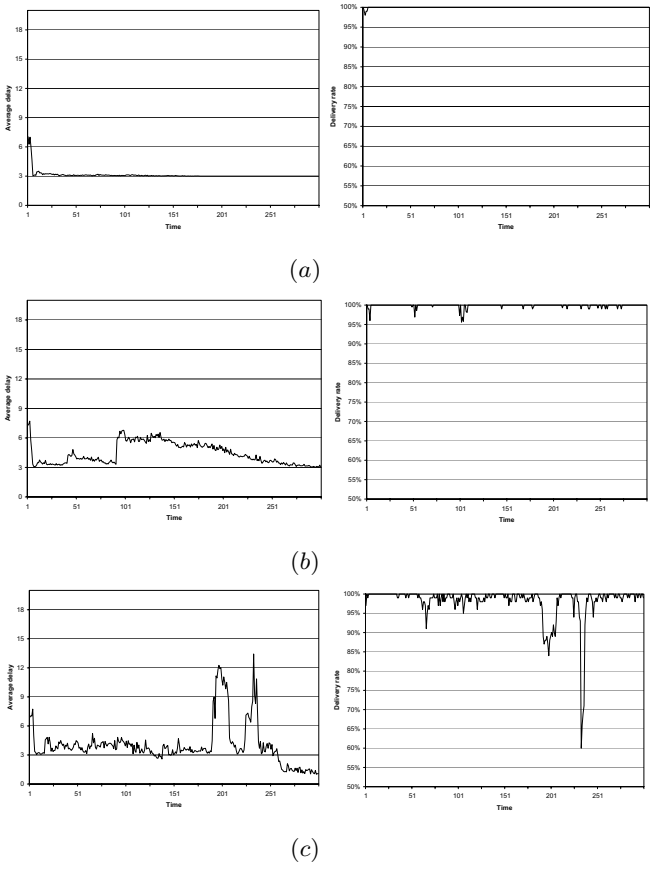

Fig. 11. Average packet delay and delivery rate of HybNet when (a) nodes do not move, (b) nodes move with speed 1 unit/clock and (c) nodes move with speed 2 unit/clock

delivery rate of HybNet when the nodes do not move is compared with the cases that nodes move with speed 1 and 2 units per clock. Regarding the fact that each node has a transmission range between 50 and 70 units, and each clock time is 1 millisecond, a speed of 2 units per clock is quite fast in the real world. The sudden drops in delivery rate happens when a path between the source and destination nodes breaks as a result of nodes movement. As the figures show, HybNet's link management strategies perform well in making the algorithm to converge to other routes and increasing the delivery rate. Since all nodes in the network including the source and destination nodes move, the length of shortest paths might vary, partly causing the fluctuations in the average delay.

4.5 The Overhead of the Algorithms

Parameters which are very important in evaluating the overhead of a routing algorithm are the size and the number of the exchanged control messages. Producing a huge number of control packets might increase the traffic in the network

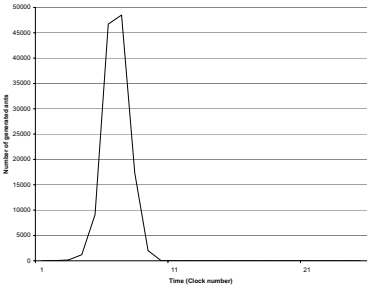

Fig. 12. Number of generated ants in ANTHOCNET at each clock time

Table 2. Total number of generated ants

Clock time	0	5	10	300
HybNet	2	6	11	301
POSANT	2.53	12.67	25.35	756.4
ANTNET	1	5	10	300
ANTHOCNET	1	10488	125048	125048

and decrease the scalability of the routing algorithm. In Figure 12 the number of generated ants by ANTHOCNET in different clock times is shown. As the graph shows, a burst of ants is generated at the beginning of route establishment when the algorithm broadcasts ants. The number of generated ants by HybNet, POSANT and ANTNET is constant in each clock time. HybNet generates n ants in each clock time (i.e. n is 1 in our experiments). ANTNET generates 1 ant in each clock time and POSANT generates at most 3 ants at each clock time. The total number of ants generated by HybNet, POSANT, ANTHOCNET and ANTNET is compared in Table 2. The total number of ants generated by ANTHOCNET grows exponentially at the beginning while it is a linear function of time in HybNet, POSANT and ANTNET. The huge number of generated ants by ANTHOCNET when it establishes a new route makes it impractical as a reactive routing algorithm. Each ant in ANTHOCNET contains a list of its visited nodes, so its size grows up as it goes far from the source node making the overhead even worse. The small number of generated ants by HybNet, POSANT and ANTNET doesn't affect the network's traffic.

5 Conclusions and Future Work

The topology of a mobile ad hoc network affects the performance of the currently existing routing algorithms. In this paper a routing algorithm called HYBNET is proposed which adapts itself with most network topologies. HYBNET uses a hybrid approach for establishing routes between a source and destination. This hybrid approach gives the algorithm a flexibility to perform well in complex

network topologies. In most cases HYBNET converges relatively fast to optimum routes using a small number of control packets. This algorithm, like most ACO routing algorithms, does not fail to establish routes when the network includes nodes with irregular transmission ranges. Our simulations confirm that HYBNET has higher delivery rate and a shorter packet delay than ANTNET, POSANT and GPSR in our network models. Also its overhead in terms of generated control traffic is much less than ANTHOCNET. Overall, HYBNET is a robust routing algorithm which performs well in mobile ad hoc networks with complex and variable topologies.

The number of parameters defined in HybNet make it difficult to apply this algorithm in real systems. A part of our future work is to address this problem by proposing mechanisms to automatically find the optimum value for each parameter. As it was shown in section 4, the performance of HybNet deteriorates when nodes move at very high speeds. Finding algorithms to enhance the performance of HybNet in such situations is another part of our future work.

References

1. Bai, F., Helmy, A.: A survey of mobility modeling and analysis in wireless adhoc networks. In: Wireless Ad Hoc and Sensor Networks (2004)
2. Bose, P., Morin, P.: Competitive online routing in geometric graphs. Theoretical Computer Science, 273–288 (2004)
3. Camp, T., Boleng, J., Wilcox, L.: Location information services in mobile ad hoc networks. In: Proceedings of IEEE Conference on Communications, vol. 5, pp. 3318–3324 (2002)
4. Di Caro, G., Ducatelle, F., Gambardella, L.M.: Anthocnet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. Special Issue on Self-Organisation in Mobile Networking 16, 443–455 (2005)
5. Di Caro, G., Dorigo, M.: Ant colonies for adaptive routing in packet-switched communications networks. In: Proceedings of the 5th ACM International Conference on Parallel Problem Solving from Nature, pp. 673–682 (1998)
6. Giordano, S., Stojmenovic, I., Blazevic, I.: Position based routing algorithms for ad hoc networks: A taxonomy. Ad Hoc Wireless Networking (2003)
7. Gunes, M., Sorges, U., Bouazizi, I.: Ara - the ant-colony based routing algorithm for manets. In: Proceedings of the 2002 International Conference on Parallel Processing Workshops, pp. 79–89 (2002)
8. Kamali, S., Opatrny, J.: Posant: a position based ant colony routing algorithm for mobile ad-hoc networks. In: ICWMC (2007)
9. Karp, B., Kung, H.T.: GPSR: Greedy perimeter stateless routing for wireless networks. In: Proc. ACM/IEEE MobiCom conference, pp. 243–254 (2000)
10. Kranakis, E., Singh, H., Urrutia, J.: Compass routing on geometric networks. In: Proc. of 11th Canadian Conference on Computational Geometry, August 1999, pp. 51–54 (1999)
11. Li, J., Jannotti, J., De Couto, D., Karger, D., Morris, R.: A scalable location service for geographic ad hoc routing. In: Proceedings of ACM/IEEE MOBICOM, pp. 120–130 (2000)
12. Yoo, J.-H., La, R., Makowski, A.: Convergence results for ant routing. In: Proceedings of CISS University of Princeton, Princeton, NJ (2004)