Adaptive Clustering Based on Auto – Learning Algorithm

Anis Ben Arbia
Prince Research Unit,
ISITC Hammam Sousse,
University of Sousse, Tunisia,
benarbia_anis@yahoo.fr

Habib youssef
Prince Research Unit,
ISITC Hammam Sousse,
University of Sousse, Tunisia
habib.youssef@fsm.rnu.tn

Abstract

This paper introduces an adaptive clustering model for wireless ad hoc networks based on Auto – Learning Algorithm (ALA). ALA allows a dynamic decomposition of the network into a virtual clusters view based on communication patterns of the mobile nodes. We consider a cluster as an Interest Group (IG) whose member nodes have common interactions. ALA is based on two types of events, New Route Events (NRE) and Route Failure Events (RFE). In this work, ALA is integrated into the well known routing protocol AODV. The adaptive version of AODV is referred to as A²ODV (Adaptive AODV). Simulation results show that A²ODV outperforms AODV with respect to packet delivery ratio, overhead, throughput, and route stability.

Keywords: *Wireless Ad hoc networks, Clustering, Routing, Auto-organization.*

1. INTRODUCTION

We consider the problem of reactive routing in wireless Ad Hoc networks. Routing refers to the activity of building forwarding tables, one for each node in the network, which tell incoming data which link to use to continue their travel towards the destination node. Communications between nodes usually pass through several intermediate hosts in which each one performs the tasks of a router. In order to be able to act as a router, mobile nodes need to use an adaptive routing protocol capable of reacting to the frequent changes of the network topology. Reactive routing algorithms such as AODV [8] discover the appropriate route as needed by broadcasting the request from the source node. The source adopts the first route reply received from the destination as the path to use for sending traffic to that destination. Such route discovery approach suffers from several drawbacks, namely (1) for large ad hoc networks, the approach results in frequent broadcast storms that may overwhelm the bandwidth and computation resources of the network, and (2) the first route reply is not necessarily the most stable or reliable route. To overcome the second problem, the source may wait for several replies to come and to select the best route with respect to desired criteria, such as number of hops, stability, etc. To overcome the first problem, clustering is one of the recommended solutions. Existing clustering solutions perform clustering using a specific criterion and allow nodes to be gathered into homogeneous clusters [1, 2, 3, 4, and 5]. Other researchers use a fixed connectivity criterion like lowest ID, one hop clusters, max-min D-cluster, k-hop clusters, etc. Each cluster is represented by one node called cluster head. A cluster head is recognized by its ID (Identifier) and elected based on various heuristics. In order to link the entire network, researchers have proposed the construction of a backbone that connects the cluster heads. We believe that clusters should not be designed based on fixed criteria (like degree, Id or diameter). Moreover, in the context of ad hoc networks, the cluster head is not necessarily equipped with additional hardware. Further it has an extra charge to assume and therefore it becomes the bottleneck of the cluster. Our Algorithm, ALA (Auto – Learning Algorithm) exploits an event model based on two types of events, New Route Events (NRE) and Route Failure Events (RFE). NREs refer to the new requests or responses in order to establish a route, and RFEs are consequences of a loss or errors due to route problems. These events reflect the communication activities in the network and reflect well, (1) the participation of a node in the routing activities of other nodes, and (2) routing needs of the nodes. Therefore, each node listens promiscuously for such events, which it uses to identify nodes that are involved in the routing paths that pass via this node. Such nodes make a

logical cluster that is dynamically maintained within the node. Future routing requests from a node are unicast to the member nodes of its cluster.

As will be shown, our algorithm is capable of creating and maintaining clusters that reflect communication patterns as well as topology changes. The objectives of the proposed clustering algorithm are twofold: (1) to reduce the number of nodes involved in the discovery of new routes, and (2) to discover more stable routes. Our approach is beneficial when the network presents clear communication patterns among the nodes. When communication is totally random, the approach is obviously counterproductive. Therefore, each node uses a metric to decide whether to launch or not the clustering algorithm. Figures 1 and 2 show a situation when clustering is beneficial and another when it is counterproductive.

The paper is organized as follows. Section 2 presents the events model. Based on this model, section 3 introduces our proposed ALA clustering algorithm. Section 4 presents simulation results. Finally, section 5 concludes the paper.

2. EVENTS MODEL

Events model is founded around Linear Temporal Logic (LTL). Below, we recall some operators of LTL.

Let S_i be a state, α and β two events; then:

$\bigcirc \beta$: β has to hold (appears) at the next state.

$\square \beta$: β has to hold on all future states.

$\diamond \beta$: β eventually has to hold (at some future state)

$\alpha \cup \beta$: β holds at the current or a future state, and α has to hold until that position. At that state α does not have to hold any more.

$\alpha R \beta$: β is true until the first state in which α is true, or forever if such a state does not exist.

FIGURE 1: A situation where clustering is solicited.

FIGURE 2: Communication needs are changing too frequently and, therefore, clustering is counterproductive.

The key issue in our design of the events model is the extraction of knowledge from a series of events recorded by each node in the network. The succession of events that have to be auto – learned are described by LTL formulas. Let $NRE_{i,j}$ ($RFE_{i,j}$) be NRE (RFE) event that involves nodes N_i and N_j .

Let,

$RREQ_{i \rightarrow j}$ be a route request event from node N_i to node N_j ,
 $RREP_{i \rightarrow j}$ a route response event from node N_i to node N_j ,
 $RERR_{i \rightarrow j}$ a route error event from node N_i to node N_j , and
 IG_i denote cluster i .

Note that $RREQ$ and $RREP$ are two NRE events, whereas $RERR$ is an RFE event.

At any node, at the occurrence of a $RREQ$, $RREP$, or $RERR$ event, at least one of the following LTL properties is verified.

1. Always whenever node N_i issues a $RERR$ to node N_j then, $N_j \notin IG_i$ and $N_i \notin IG_j$

$$\frac{(\Box RERR_{i \rightarrow j})}{N_j \notin IG_i \text{ and } N_i \notin IG_j}$$

2. Node N_j never sends a $RREP$ to node N_i until node N_i issues a $RREQ$ to node N_j , then, $N_j \notin IG_i$ and $N_i \notin IG_j$.

$$\frac{(\Diamond RREQ_{i \rightarrow j}) \mathcal{R} (\Diamond \neg RREP_{j \rightarrow i})}{N_j \notin IG_i \text{ and } N_i \notin IG_j}$$

3. Always whenever node N_j issues a $RERR$ (or never sends a $RREP$) to node N_i until N_i issues a $RREQ$ to node N_j . Then, $N_j \notin IG_i$ and $N_i \notin IG_j$

$$\frac{(\Diamond RREQ_{i \rightarrow j}) \mathcal{U} (\Box \neg RREP_{j \rightarrow i}) \vee (\Box RERR_{j \rightarrow i})}{N_j \notin IG_i \text{ and } N_i \notin IG_j}$$

4. Whenever N_i issues a $RREQ$ to nodes N_j to which node N_j responds with a $RREP$, then $N_j \in IG_i$ and $N_i \in IG_j$

$$\frac{\Box ((\Diamond RREQ_{i \rightarrow j}) \mathcal{U} (\Box RREP_{j \rightarrow i}))}{N_j \in IG_i \text{ and } N_i \in IG_j}$$

5. Whenever N_i issues a $RREQ$ to node N_j to which node N_j responds with a $RREP$, and no $RERR$ is initiated by N_j to node N_i , then $N_j \in IG_i$ and $N_i \in IG_j$

$$\frac{\Box ((\Diamond RREQ_{i \rightarrow j}) \mathcal{R} (\Diamond RREP_{j \rightarrow i}) \wedge \Box \neg RERR_{j \rightarrow i})}{N_i \in IG_i \text{ and } N_j \in IG_j}$$

6. Always whenever a RERR is initiated by node N_i to node N_j then, $N_j \in IG_i$ and $N_i \in IG_j$.

$$(\Box \neg RERR_{i \rightarrow j})$$

$$N_j \in IG_i \text{ and } N_i \in IG_j$$

The Auto Learning Algorithm (ALA) is executed at each node and consists mainly of the verification of the above LTL formulas. The execution of ALA is triggered by the occurrence of a routing event. Based on the outcome of this evaluation, each node N_i computes a coefficient μ_{ij} for each of the nodes N_j involved in this event. This coefficient takes values in the range [0; 1], where values greater than or equal to 0.5 mean that the two nodes belong to each other IG.

The coefficient μ_{ij} is computed as follows,

$$\mu_{ij} = \sigma \alpha_{Li \rightarrow j} + (1 - \sigma) \alpha_{Ti \rightarrow j}$$

Where,

$$0 \leq \sigma \leq 1.$$

$\alpha_{Li \rightarrow j}$ is computed based on the six formulas defined in Section 2 as indicated in Table 1, and

$\alpha_{Ti \rightarrow j}$ (coefficient of use in time) is computed based on rate of occurrence of routing events as explained in Figure 4.

Table 1 gives the numerical values assigned to α_L when each of the LTL formulas is evaluated. Each of these values is in the range [0;1] and is equal to the ratio of the number of times the formula evaluated to true to the total number of times that formula has been evaluated, from $t=0$ until now. Let $\alpha_{Li \rightarrow j}(n)$ be the value assigned to $\alpha_{Li \rightarrow j}$ as a result of the evaluation of formula (n), $n = 1, 2, \dots, 6$, when node N_i receives a routing event involving N_j . Then, the reader can easily verify that the following is true:

$$\alpha_{Li \rightarrow j}(1) + \alpha_{Li \rightarrow j}(6) = 1$$

$$\alpha_{Li \rightarrow j}(2) + \alpha_{Li \rightarrow j}(4) = 1$$

$$\alpha_{Li \rightarrow j}(3) + \alpha_{Li \rightarrow j}(5) = 1$$

Formulas	Updating value of $\alpha_{Li \rightarrow j}$
Formula -1-	0
Formula -2-	$\frac{\sum_{t.time} RFE_{j \rightarrow i}}{\sum_{t.time} (NRE_{j \rightarrow i} + RFE_{j \rightarrow i})}$
Formula -3-	$\frac{\sum_{t.time} (RFE_{j \rightarrow i} + RFE_{i \rightarrow j})}{\sum_{t.time} (NRE_{i \rightarrow j} + NRE_{j \rightarrow i} + RFE_{i \rightarrow j} + RFE_{j \rightarrow i})}$
Formula -4-	$\frac{\sum_{t.time} NRE_{j \rightarrow i}}{\sum_{t.time} (NRE_{j \rightarrow i} + RFE_{j \rightarrow i})}$
Formula -5-	$\frac{\sum_{t.time} (NRE_{j \rightarrow i} + NRE_{i \rightarrow j})}{\sum_{t.time} (NRE_{i \rightarrow j} + NRE_{j \rightarrow i} + RFE_{i \rightarrow j} + RFE_{j \rightarrow i})}$
Formula -6-	1

Table 1: Updating of the values of α_L as soon as formula (s) is (are) checked.

The second coefficient α_{Ti} estimates the occurrence intensity of routing events within the node. Figure 3 shows a diagram explaining how α_T is computed. The figure shows a sequence of events consisting of a $RREQ_{ij}$, followed by a $RREP_{ji}$, $RREP_{mi}$, $RREP_{li}$, and finally a $RERR_{ki}$. Let t_{ij} , t_{ji} , t_{mi} , t_{li} , and t_{ki} be the instants at which each of these events has occurred.

For a $RREP_{ji}$ to $RREQ_{ji}$

$$\alpha_{Tij} = [1 - (t_{ji}/\text{Now})]$$

For $RERR_{ki}$ to $RREQ_{ij}$:

$$\alpha_{Tik} = [t_{ki} - t_{ij}] / \text{Now}$$

FIGURE 3: Updating of α_T by node i.

The formulas in Figure 3 closely depend on the routing events occurrence (NREs, RFEs) as well as the time between these events.

3. CLUSTERING TERMINOLOGY

3.1 Clustering based ALA

Recall that each mobile node N_i acts as a router. To apply our model, we have added to the routing table of each node i, a new field indicating for each next hop j the value of μ_{ij} . Recall that μ_{ij} tells node N_i whether N_j belong to IG_i or not.

At the beginning, every node forms an IG by itself. IGs start appearing after a time LT (Learning Time), which represents the time needed to learn about the activities of the network. LT is chosen such that the clusters (IG) cover more than $N/4$ of the nodes, where N is the number of mobile nodes in the network. A possible future extension should define LT depending on the events model.

As new routes are established and existing routes break down, IGs get modified by the nodes involved in these routing activities. Figure 4 shows an example of a scenario where node 1 receives and classifies events according to the nodes involved in each event. Nodes calibrate their μ_{ij} when necessary and update their routing tables. ALA uses these values to detect similarities between nodes. For example, in Figure 5, node N_1 and node N_2 have $\mu_{12}=0.76(>0.5)$, $\mu_{21}=0.7(>0.5)$ ($\sigma=0.5$), N_1 and N_2 will join a same IG.

FIGURE 4: Clustering based events.

			N ₂		
			Destination@	Next Hop	μ
			N ₁	P	0.7
			R	S	0.27
			O	N	0.23
			N ₁		
			Destination@	Next Hop	μ
			N ₂	Y	0.76
			Z	K	0.32
			U	M	0.12

FIGURE 5: ALA mechanism

3.2 The algorithm ALA

In the following, we present the ALA algorithm:

IG : Interest Group

N_{id} : Node Identifier

IG_{id} : Interest space identifier

my_IG: IG of the local node

//Adding a node to my_IG

```

On Deciding IG IG_Update (Nid)
    { if ( $\mu_{Me,id} \approx \mu_{id,Me}$ ) and  $\mu_{Me,id} > 0,5$ 
      {
        my_IG = my_IG  $\cup$  {Nid}
        for (Nid in my_IG)
          Multicast (my_IG)
        }
      }

```

//Deleting a node from my_IG

```

On deleting my_IG IG_Request (Nid)
    { for (Nid in my_IG)
      {
        if ( $\mu_{Me,id} \neq \mu_{id,Me}$ )
          {
            my_IG = my_IG  $\setminus$  {Nid}
            for (Nid in my_IG)
              Multicast (my_IG)
            }
          }
      }

```

//On receiving a request to update of my_IG

```

On Receiving IGid IG_Request (Nid)
    {
      my_IG = my_IG  $\cup$  {Nid}
    }

```

To illustrate the ALA mechanisms, we propose the example showed in figure 6. Suppose that Node 8 ($\in IG_1$) wants to add node 14 to its IG; the steps to do are:

→ On Deciding IG Update (*N₁₄*)

----- if conditions are fulfilled, then each node in IG_1 receives a request to update its table and performs:

→ Receiving IG_1 $IG_Request$ (N_1)

----- N_2, N_4, N_6, N_{10} and N_{12} update their IGs .

Suppose, now, that Node 7 ($\in IG_3$) wants out of its IG ; step to do is:

→ On deleting my IG $IG_Request$ (N_7)

----- N_1, N_3, N_5, N_9 update their IGs (N_7 is out)

FIGURE 6: Example of ALA mechanism

4. SIMULATION RESULTS

4.1 Clustering experimentations

We simulate the clustering algorithm by placing N nodes on a 100 x 100 area. We assume two nodes can hear each other if their distance is within a predefined transmission range. We assume an ideal network model where a link can be established between any two nodes within transmission range of each other, or when a path between the two nodes can always be found by the routing algorithm. The purpose of the first set of experiments is to show that adaptive clustering based on Auto–Learning Algorithm is capable of clustering the nodes based on observed communication patterns. The number of nodes is $N = 12$, each four nodes are placed separately on each side of the area (it's like four sub-networks separated), each node moves along the side of the area. Communications permitted are as follows:

First between N_1, N_2, N_3 and N_4 , second between N_1, N_5, N_6 , and N_7 , then between N_7, N_8, N_9 and N_{10} , and finally between N_3, N_{10}, N_{11} and N_{12} . We trace all events attempted between nodes (RREQ, RREP, RERR) and extract the values of μ for each node with ($\sigma=0.5$) (Table. 2). As expected, the network is clustered into 4 clusters: $IG_1 = \{N_1, N_2, N_3, N_4\}$, $IG_2 = \{N_1, N_5, N_6, N_7\}$, $IG_3 = \{N_7, N_8, N_9, N_{10}\}$ and $IG_4 = \{N_3, N_{10}, N_{11}, N_{12}\}$ (see Figure 7).

N	1	2	3	4	5	6	7	8	9	10	11	12
1	1	0,81	0,69	0,57	0,63	0,71	0,68	0	0	0	0	0
2	0,81	1	0,72	0,58	0	0	0	0	0	0	0	0
3	0,69	0,72	1	0,86	0	0	0	0	0	0,56	0,54	0,59
4	0,57	0,58	0,86	1	0	0	0	0	0	0	0	0
5	0,63	0	0	0	1	0,69	0,81	0	0	0	0	0
6	0,71	0	0	0	0,69	1	0,74	0	0	0	0	0
7	0,68	0	0	0	0,81	0,74	1	0,69	0,64	0,52	0	0
8	0	0	0	0	0	0	0,69	1	0,91	0,63	0	0
9	0	0	0	0	0	0	0,64	0,91	1	0,71	0	0
10	0	0	0,56	0	0	0	0,52	0,63	0,71	1	0,82	0,68
11	0	0	0,54	0	0	0	0	0	0	0,82	1	0,71
12	0	0	0,59	0	0	0	0	0	0	0,68	0,71	1

Table 2: μ with $\sigma=0.5$

FIGURE 7 : Clustering results

FIGURE 8: Number of IGs in the Time (Sc1, Sc2)

4.2 A²ODV: AODV based ALA

The purpose of the second set of experiments is to show that adaptive clustering based on Auto–Learning Algorithm improves the performance of the underlying reactive routing protocol (AODV in our case). In AODV, when a node source desires to get a path to a destination, for which it does not already have a route, it broadcasts a Route Request (RREQ) packet across the network. Nodes receiving this RREQ packet update their routing information for the source node. The RREQ contains the most recent sequence number for the destination of which the source node is aware. A node receiving the RREQ may send a Route Reply (RREP) if it is either the destination or an intermediate node which has a route to the destination with a sequence number greater than or equal to that contained in RREQ. If this is the case, it propagates back a RREP to the source. AODV relies on routing table entries to propagate a RREP back to the source and subsequently, to route data packets to the destination. When a link breaks down, the concerned node invalidates in its routing table all destinations that become unreachable due to the loss of the link. It then creates a Route Error (RERR) message in which it lists each of these lost destinations. The node sends the RERR upstream towards the source node. If there are multiple previous hops (precursors) which utilize this link, the node will broadcast the RERR; otherwise, it is unicast. When a node receives a RERR, it first checks whether the node that sent the RERR is its next hop to any of the destinations listed in the RERR. If it is the case, the node invalidates these routes in its route table and then propagates the RERR back towards the source. Once the source receives the RERR, it can reinitiate route discovery if it still requires the route.

AODV operations are based on four actions: 1- Sending RREQ, 2- Receiving RREQ, 3- Retransmitting RREP and 4- Receiving RREP. We have modified each of these actions to take into account ALA algorithm. Due to limited space, we only present updates made to the first two actions:

```

-----SENDING RREQ
if (no route exists)
{
    check request already sent for destination
    if (no request sent already)
    {
        create a RREQ packet
        add (dest addr, IGid) to request buffer

//RREQ to my IG
        locally multicastIGid RREQ
        set timer for RREP_WAIT_TIME before

//RREQ to the rest of the network
        locally multicast to the rest
        set timer for RREP_WAIT_TIME before
    }
    else
        buffer packet from stream or discard
}
}

```

Based on ALA, RREQ will be sent first to my IG. If there is no RREP; in that case it will be sent to the other IGs including a node from my IG (N1, N3, N7, N10 in Figure 6), and if necessary, to the rest of the network.

```

-----RECEIPTING RREQ
On receiving RREQ
{
    {
        check the type of the RREQ
        if it's a multiastfrom_my_IG
        forward to all my_IGs
        add (source addr, multiastfrom_my_IG) to request buffer
        elseif ( (source addr, multiastfrom_my_IG) in request buffer )
        {
            delete request //already heard and processed
        }
        else
            add (source addr, multiastfrom_my_IG) to request buffer
        }
    }

if (no route to the source)
    create a route entry for source addr
elseif (source seq_number in RREQ > source seq_number in route entry)
    update route entry for source addr
else ((source seq_number in RREQ = source seq_number in route entry) AND (hop count in RREQ < hop
count in route entry))
    update route entry for source addr
    if (I'm the destination of RREQ)
    {
        create a RREP packet
        unicast RREP to the source of the RREQ
    }
    else
        forward RREQ
}

```

If the above algorithm is running in a node included in more than one IG and the type of the RREQ is *multiast_{from_my_IG}*, then the node multicasts it to all of its IGs.

Next we present simulation results of AODV and A²ODV. The simulation platform is NS2. The mobility model uses a random waypoint model in a square area 1000x1000 m². The sources traffic generators are CBR (Continuous Bit Rate). The packet sizes are 512 bytes. We simulate 100 nodes identified by numerical identifiers. The speed of each node is uniformly distributed between 4 and 15 m/s. In our case, mobility model is specified without stationary intervals. To observe the ALA results, we have conceived two scenarios; Sc1 is an Ad hoc network totally random with no remarkable interactions between nodes, which is unfavourable to the formation of IGs. On the other hand, in Sc2 the network is composed of nodes working commonly, and in that case, ALA should be beneficial. Figure 8 shows the number of IGs as time progresses for Sc1 and Sc2. As shown in Figure 8, Sc2 is a scenario which represents a good situation where ALA is performing well. Therefore, in the following experimentations, we simulate A²ODV and AODV under Sc2.

Figure 9 compares A²ODV and AODV in terms of routes declared down. This parameter indicates the stability of the routes used by the protocol. And as it is clear in Figure 9, A²ODV decision is right. Other comparisons are made with respect to packet delivery ratio, overhead, and throughput. In A²ODV, the network is composed of IGs each containing nodes that can cooperate to choose the best routes. The requests of nodes become predictable events. The cooperation between nodes, which are distributed into IGs, facilitates the operations in the routing tables especially in the route decision procedures and localizations. A²ODV performs well in correlation with the dynamics of the network. The first results (Figure10) show a large profit in terms of packet delivery ratio and overhead due to the stability of the chosen routes and the limited number of RREQ, RREP packets. RREQ flooding is henceforth selective and the RREP are generated by only a limited number of nodes depending on the IGs to which they belong. This is needed to limit the control traffic flow (Figure 12) and optimize the protocol throughput performance (Figure 11).

FIGURE 9 : Route changes A²ODV vs AODV

FIGURE 10: PACKET DELIVERY RATIO

FIGURE 11: Throughput A²ODV vs AODV

FIGURE 12: Overhead A²ODV vs AODV (SC2)

5. CONCLUSION

This paper introduces an adaptive clustering approach for wireless ad hoc networks based on Auto – Learning Algorithm (ALA). ALA allows a dynamic decomposition of the network into a virtual clusters view based on routing activities of the mobile nodes. Routing activities observed by each node are described using LTL formulas. ALA has been integrated in the AODV routing protocol. The modified AODV is called A²ODV. Simulation results show, that when communications are not totally random, A²ODV outperforms AODV with respect to throughput, packet delivery, and overhead.

REFERENCES.

- [1] A.L.N. Fred and A.K. Jain, "Data Clustering Using Evidence Accumulation", Proc. of the *16th Intl. Conf. on Pattern Recognition, ICPR 2002*, Quebec City, pp. 276 – 280, 2002.
- [2] A. Strehl and J. Ghosh, "Cluster ensembles - a knowledge reuse framework for combining multiple partitions". *Journal on Machine Learning Research*, 3, pp. 583-617, 2002
- [3] A. Topchy, A.K. Jain, and W. Punch, "Combining Multiple Weak Clusterings", Proc. 3d *IEEE Intl. Conf. on Data Mining*, 331-338, 2003
- [4] X. Fern, and C. E. Brodley, "Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach", In Proc. *20th Int. conf. on Machine Learning, ICML 2003*.
- [5] A.K. Jain and J.V. Moreau, "The Bootstrap Approach to Clustering", in *Pattern Recognition Theory and Applications*, P.A. Devijver and J. Kittler (eds.), Springer-Verlag, 1987, pp. 63-71.
- [6] TURGUT D., TURGUT B., ELMASRI R., «Optimizing clustering algorithm in mobile ad hoc networks using simulated annealing». *WIRELESS COMMUNICATIONS AND NETWORKING CONFERENCE (WCNC), NEW ORLEANS, USA, MARCH 2003*.
- [7] WONG Y., WONG C., «A Fuzzy decision based routing protocol for mobile ad hoc networks». *Actes du 10th IEEE International Conference on Networks, 2002, ICON 2002, Singapore*.
- [8] C. E. PERKINS E. M. ROYER, "AD-HOC ON-DEMAND DISTANCE VECTOR ROUTING," IN PROCEEDINGS OF THE IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS, NEW ORLEANS, LA, FEBRUARY 1999, PP. 90–100.