An Efficient Cluster-Based Routing Algorithm in Ad Hoc Networks with Unidirectional Links*

YI-YU SU, SHIOW-FEN HWANG AND CHYI-REN DOW
Department of Information Engineering and Information Science
Feng Chia University
Taichung, 407 Taiwan

Mobile ad hoc networks are dynamically organized by a collection of wireless mobile nodes. The mobile nodes in ad hoc networks can move arbitrarily thus the topology of network changes dynamically. Due to the properties of communication medium in wireless networks, unidirectional links may exist between mobile nodes thus results in the difficulty of link utilization and routing. In this paper, we take the advantages of multi-hop acknowledgement and employ the clustering technique to design an efficient hybrid routing protocol in ad hoc networks with unidirectional links. Simulation results demonstrate the stability and efficiency of the proposed protocol.

Keywords: mobile ad hoc networks, unidirectional link, routing, clustering, efficient, hybrid

1. INTRODUCTION

Ad hoc networks are a collection of wireless mobile nodes forming a temporary wireless network. In ad hoc networks, any pre-established wired or wireless infrastructure and the centralized administration, such as the base stations, are unavailable. Mobile nodes can move arbitrarily results in the constant changing topology. Thus, ad hoc networks are specially important and useful in battlefield or disaster area. Under the limited resources such as network bandwidth, memory capacity, and battery power of the mobile nodes, the efficiency of routing protocols in ad hoc networks becomes more important and challenging. In ad hoc networks, current routing protocols can be classified into proactive (table-driven) [36], reactive (on-demand) [6, 18, 22, 29, 30] or hybrid routing [13, 14, 20, 27, 37]. Each node of the proactive routing scheme periodically maintains one or more tables to store consistent and up-to-date routing information from one to every other node in the network. On the contrary, reactive routing scheme constructs and maintains routes only when desired by a source node. In medium to large size ad hoc networks, the former scheme requires to update topology information globally and to exchange routing table periodically, which incurs large control overhead against routing efficiency. The later scheme suffers from longer packet delay caused by route discovery and network-wide overhead of route query flooding.

To balance the performance and overhead, hybrid routing scheme is proposed to take the advantage of the above two schemes. The zone routing protocol (ZRP) [13, 14, 20, 27, 37] is an example of hybrid routing scheme. In ZRP, each node periodically maintains routing information only within local region. While the route to the destination

Received November 21, 2006; revised July 3, 2007; accepted October 9, 2007.
Communicated by Ten-Hwang Lai.

is unavailable in the local routing table, the route query is border-casting [14] to the network for route discovery. On the one hand, it limits the scope of proactive procedure, and thus incurring less overhead comparing with proactive schemes. On the other hand, the global route discovery through the network is performed by efficiently querying selected nodes (2-hop neighbor of each node), which reduces the route request flooding comes of reactive schemes.

Similar to hybrid routing schemes, clustering technique is proposed [2, 3, 5, 7, 8, 10, 11, 15, 17, 19, 23, 31, 41, 42] to improve routing performance and to benefit from both reactive and proactive schemes. By using the clustering algorithm, all nodes in the network are divided into clusters, thus a large network can be virtually partitioned into several sub-networks. In each cluster, the cluster head maintains its cluster members and intra-cluster topology information. Cluster members that connect to adjacent clusters are gateway nodes. The cluster based routing scheme further improves the performance of conventional hybrid routing scheme. Since only cluster heads maintain local topology information, the overhead of exchanging route table of each node is reduced. Additionally, route discovery among clusters can be done by efficient broadcasting of cluster heads and gateway nodes, network-wide route request flooding is prevented in this way. Many clustering algorithms have been proposed, such as the Lowest-ID Clustering [15], High-Connectivity Clustering [15], Weighted Clustering [5, 8, 10], *etc.* [2, 3, 7, 11, 17, 19, 23, 39, 41]. CBRP [17] is an example of cluster based routing protocol, which employs Lowest-ID clustering algorithm to divide nodes into a number of overlapping or disjoint 2-hop-diameter clusters. By making use of 2-hop topology information maintained by each node, an active route between source and destination can be shortened dynamically by intermediate forwarding nodes. Moreover, a broken route may be repaired locally thus avoiding the cost of rediscover a new route.

However, most of the existing clustering algorithms usually result in lots clusters due to mobility of nodes and inefficient maintenance of cluster, which diminish the advantages of the cluster structure. Since the more clusters in the network, the more nodes may act as gateway nodes and cluster heads that incur more maintenance overhead and more periodical control packets. Besides, some routing protocols and clustering algorithms assume that all nodes have the same transmission range and all links are bidirectional in the network. In fact, this does not reflect the real environment where the characteristics of radio propagation in wireless network may produce unidirectional links which are originated from radio interference, hidden terminal problem, or differences in transmission range caused by power consumption or heterogeneous devices. Link asymmetry may cause inefficient or abnormal of most routing protocols that considers only bidirectional links, and undiscovered unidirectional links represents untapped network capacity and thus reducing network connectivity [1, 4, 9, 12, 16, 21, 24-26, 28, 32-35, 38, 40, 43-45]. Accordingly, in this paper, we take the advantage of multi-hop acknowledgement and employ the clustering technique to design an efficient hybrid routing protocol in ad hoc networks with unidirectional links. We also propose the maintenance scheme that ensures our clustering and routing perform stably in mobile environments.

The rest of this paper is organized as follows. We describe the proposed clustering algorithm and cluster maintenance scheme in section 2. The routing protocol and route maintenance is proposed in section 3. The simulation results are presented in section 4. Finally, the conclusions and some further improvement are offered in section 5.

2. CLUSTERING ALGORITHM

In this section, we introduce our clustering algorithm and routing protocol. First, we discuss the link management of our scheme, and then we propose the clustering algorithm. Finally, the maintenance scheme is discussed which adapts our clustering algorithm to dynamic topology.

2.1 Link Management of Unidirectional Links

In ad hoc networks with only bidirectional links, mobile nodes use Hello and I-Hear-You to detect neighbor nodes within transmission ranges. If the transmission ranges of nodes were different, the network may have unidirectional links and the traditional link layer acknowledgement may not be able to detect the unidirectional links correctly. Fig. 1 is an example of a unidirectional link from node *x* to node *y*. Since the transmission range of node *y* is smaller than node *x*, node *x* can not receive the acknowledgement from node *y*. In this case, without the support of any other techniques, only node *y* knows this unidirectional link.

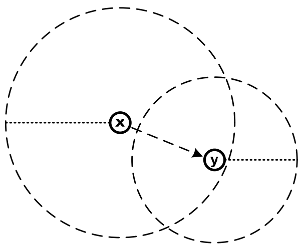


Fig. 1. Transmission range induced link asymmetry.

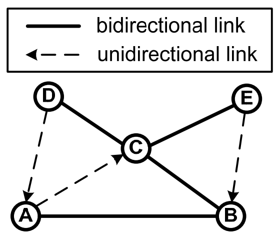


Fig. 2. Multi-hop link layer acknowledgement.

In order to make use of unidirectional links, we employ tunneling [26] and multi-hop acknowledgement [28] techniques to detect the unidirectional links and maintain relationships of unidirectional neighbors. First, we define the time-to-live (TTL) of multi-hop acknowledgement packets, which enables the acknowledgement to be re-broadcasted by neighbor nodes for further detection of the existence of unidirectional links. We illustrate this approach by Fig. 2, node A broadcasts a Hello packet to its neighbors, but node C cannot reply an acknowledgement (ACK) to node A directly, because the unidirectional link is from node A to C. Thus, node A may lose its unidirectional neighbor node C and this link becomes useless and even is harmful to routing protocol [43]. By using the tunneling approaches and multi-hop acknowledgement, the ACK of node C to node A is re-broadcasted to its neighbors within TTL-hops. In this way, the ACK can be transmitted back to node A via node D or B (while TTL is 2). At the same time, the path traveled by this ACK is piggybacked by each node. Afterward, node A receives this ACK, and then, it sends a new HELLO packet which includes the path C-D-A or C-B-A that depends on which ACK packet arrived node A first. Therefore, node C knows the path to its unidirectional neighbor node A, such as path C-D-A or C-B-A. Hereafter, node A and node C know the existence of each other and the path between node A and C is established. In this example, node A knows the path back to node D as well. As a result, the existence of unidirectional links between nodes can be detected and the link management of nodes that detect unidirectional links is accomplished. We set TTL = 2 to achieve basic unidirectionality with minimum additional cost of the network.

2.2 Clustering Algorithm

Initially, we define the cluster information that is maintained by each node. The cluster information is very important that keeps track of the necessary information for our clustering algorithm, which will be described in the following section. After updating the cluster information, a node can determine its own status easily by exchanging the cluster information with its neighbor nodes. The cluster information is also used for cluster maintenance and routing. Before we introduce the clustering algorithm, some notations are defined in Table 1.

Table 1. Notation definition.

Notation	Descriptions
V	The set of all nodes.
$N_1(v)$	The set of 1-hop neighbors of node v.
$N_2(v)$	The set of 2-hop neighbors of node v.
$d(u, v)$	The least hops from node u to node v.
$label(v) = U$	The status that node v is undetermined.
$label(v) = D$	The status that node v is a member of a cluster.
$label(v) = H$	The status that node v is a cluster head of a cluster.
pendant node	A node has only one neighbor which is bidirectional linked.

$$s(v) = \left|\{x \mid x \text{ is a pendant node} \,\&\, x \in N_2(v)\}\right| \quad (1)$$

$$t(v) = \left|\{x \mid label(x) = U \,\&\, x \in N_2(v)\}\right| \quad (2)$$

$s(v)$ is the number of pendant nodes in v's 2-hop neighbors, $t(v)$ is the number of nodes which status is U in v's 2-hop neighbors, both of them will be used in the defined weight function. Each node maintains its neighbor tables that include bidirectional neighbor table and unidirectional neighbor table. The node's ID, degree, status and current cluster head are maintained as a list in both bidirectional and unidirectional neighbor table. The path towards each unidirectional neighbor is also maintained in unidirectional neighbor table. The cluster information includes $id(v)$, $dom(v)$, $deg(v)$, $hth(v)$, $next(v)$, $ncm(v)$, $gw(v)$, as shown in Table 2. Note that the neighbor table and cluster information require just 1-hop neighbor information. Nodes do not periodically maintain 2-hop neighbor information, except for competing to become a cluster head (a node with status U and cannot join any other cluster in its neighbors).

Before executing the clustering algorithm, a node with status U first exchanges messages within its neighbors to build the neighbor tables and collects the cluster information. Then, each node p executes the clustering algorithm as shown in Fig. 3.

Table 2. Cluster information definition.

Notation	Descriptions
id(*v*)	The ID of node *v*.
dom(*v*)	The cluster head of node *v*.
deg(*v*)	The degree of node *v*.
hth(*v*)	The minimum hops to the cluster head of node *v*.
next(*v*)	The ID of the next hop node to the cluster head of node *v*.
ncm(*v*)	The number of the cluster members of nodes *v*'s cluster.
gw(*v*)	TRUE, if node *v* is a gateway node of its cluster, FALSE, otherwise.

```
if    label(p)=U
      {
      if    ∃ y ∈ N_1(p) and (label(y) = H or hth(y) = 1)
      {
          label(p)=D;
          hth(p)=hth(y)+1;
          next(p)=y;
          dom(p)=dom(y);
      }
      else
      {
          w(p) = 0.7 × s(p) + 0.2 × t(p) + 0.1 × deg(p);
          if    w(p) > w(x), ∀x ∈ N_1(p)
          {
               become_head(p);
          }
          else  let q be the node with  w(q) = max_{∀x∈N_1(p)} w(x)
          {
               if    w(q) > w(x), ∀x ∈ N_1(q)
               {
                    become_head(q);
               }
               else  let r be the node with  w(r) = max_{∀x∈N_1(q)} w(x)
               {
                    become_head(r);
               }
          }
      }
}
```

Fig. 3. The clustering algorithm.

```
become_head(x)
{
     label(x)=H;
     hth(x)=0;
     ∀ y ∈ {p | p ∈ N_1(x) & label(p) = U}
     {
          label(y)=D;
          hth(y)=1;
          dom(y)=x;
          next(y)=x;
          ∀ z ∈ {q | q ∈ N_1(y) & label(q) = U}
          {
               label(z)=D;
               hth(z)=2;
               dom(z)=x;
               next(z)=y;
          }
     }
}
```

Fig. 4. *become_head* procedure of the clustering algorithm.

In the clustering algorithm, the radius of the cluster is 2-hop and we ensure that each member node (status is *D*) belongs to a cluster head which is one of its 2-hop neighbors. The clustering algorithm is fully distributed and it starts at any node which status is *U*. If a node with status *U* can not join any neighbor clusters, it calculates its weight by exchanging 2-hop neighbor information, and compares its weight with 1-hop neighbors that status is *U*, to determine the status of each other. For each node *p*, the weight function *w*(*p*) is defined as:

$$w(p) = (0.7 \times s(p) + 0.2 \times t(p) + 0.1 \times deg(p)). \tag{3}$$

$s(p)$ is the number of pendant nodes (defined in Table 1) in node p's 2-hop neighbors, $t(p)$ is the number of nodes which status is U in node p's 2-hop neighbors, and $deg(p)$ is the degree of node p. Note that we define the degree of a bidirectional link equals to 1 and a unidirectional link equals to 0.5. For example, in Fig. 5, the weight of node 1, node 3, and node 7 are:

$$w(1) = (0.7\times1+0.2\times4+0.1\times1) = 0.7+0.7+0.1 = 1.6,$$
$$w(3) = (0.7\times2+0.2\times10+0.1\times4) = 1.4+2.0+0.4 = 3.8,$$
$$w(7) = (0.7\times2+0.2\times9+0.1\times5) = 1.4+1.8+0.5 = 3.7.$$

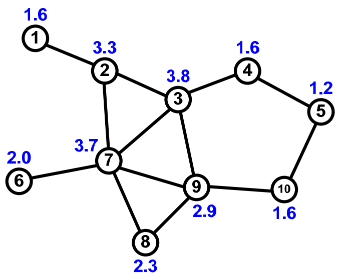


Fig. 5. Weight of nodes with status *U*.

After the weight function calculation is finished, each node compares the weight with 1-hop neighbors for status determination. We design a two rounds neighbor competition scheme to complete the status determination of each node. For easy understanding, we take Fig. 5 as an example to illustrate the procedure. In Fig. 5, assuming that node 10 starts the competition procedure, it first compares its weight with node 5 and 9(1-hop neighbors of node 10). If node 10 has the largest weight in its 1-hop neighbors at first round of the competition, it would become a cluster head and the competition process for node 10 is completed. But in this example, the status of node 9 is *U* and it has larger weight than node 10. Thus, node 9 wins the first round weight competition, and it becomes a candidate node for being a cluster head (informed by node 10 which started the competition procedure). Then, node 9 compares its weight with node 3, 7, 8 and 10 (1-hop neighbors of node 9). Again, if node 9 has the largest weight in its 1-hop neighbors at the second round of the comparison, it would become a cluster head and the competition procedure is completed. However, in this example, node 3 wins because it has larger weight and it would be informed by node 9 to become a cluster head. The entire competition procedure of node 10 is finished. The rest nodes in this example will become the cluster member of node 3 (Fig. 6). In fact, in this example, if all nodes start with status *U*, no matter which node begins the clustering procedure, after finishing the competition procedure, node 3 will always become the cluster head and other nodes will join this cluster.

Fig. 7 is an example of clustering with unidirectional links. The only difference is the weight of each node may be different because of the differences of the weight between unidirectional links and bidirectional links.

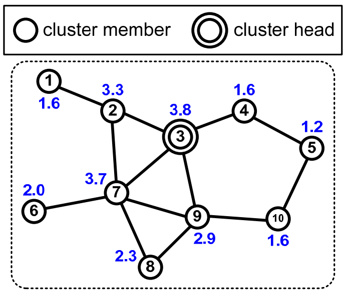


Fig. 6. An example of the clustering algorithm.

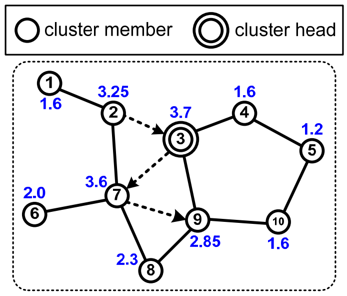


Fig. 7. Clustering with unidirectional links.

In most of the existing clustering algorithm, the clusters formed at the border of the network usually have less cluster members compare with the clusters at the center of the network. Since the degree of nodes at the border of the network is usually less. This may result in more small clusters after clustering, which increases the number of cluster heads and gateway nodes, and incurs more route request and maintenance overhead as well. In the proposed clustering algorithm, we consider the degree of the node and the number of nodes can be managed by a cluster head, since both of them are important in most clustering algorithms. In addition, the number of pendent node is also considered and integrated into the weight function. If a node has more pendant nodes within its 2-hop neighbors, it would have higher possibility to become a cluster head. In this way, the cluster heads formed by (taking care of) pendant nodes can be reduced, and thus improving the efficiency of cluster structure. The performance results of different weight function considerations are shown in Section 4. Moreover, the proposed weight function requires only status and degree of nodes, which can be obtained directly and easily, and is as simple as Lowest-ID or High-Connectivity clustering algorithm. Comparing with existing weight based clustering algorithms [5, 8, 10], the mobility information, battery power, distance between nodes, and transmission rate are required for weight calculation of each node, which detracts from the generality of the clustering algorithm and may not always be available for different mobile devices.

Overall, our clustering algorithm has several advantages:

1. Pendant nodes are usually not cluster heads, which ensures better stability and higher routing performance of the cluster based routing.
2. The number of clusters in the network will be reduced by employing our two rounds weight competition scheme. This improves the stability of cluster and less maintenance overhead comparing with the existing clustering algorithms that generate more small clusters.
3. With proper link management, after clustering, the unidirectional links are useful for routing.

2.3 Cluster Maintenance Scheme

In ad hoc networks, topology changes frequently due to node mobility. We propose a cluster maintenance scheme to ensure the network always clustered well. The situations will be considered as follows.

2.3.1 New link caused by a new node in the network

A new node *v* may join the network after the clustering procedure is finished. Node *v* would collect the cluster information and choose the node with the smallest *hth*() and smaller than 2 from its 1-hop neighbors, then joins the cluster. If no such node exists, node *v* starts the clustering algorithm and may become a cluster head.

2.3.2 New link between clusters

When there is a new link between nodes *u* and *v*, *dom*(*u*) and *dom*(*v*) are different, one of these two nodes may join the other's cluster if the *hth*() could become smaller after joining the cluster, or nodes *u* and *v* would become the gateway nodes of their clusters without changing the cluster head. After that, nodes update the cluster information.

2.3.3 New link in a cluster

When a new link occurs between two nodes in a cluster, the *hth*() and *next*() may change due to the new link, nodes should update the cluster information which ensures the cluster head knows the up-to-date link state in the cluster.

2.3.4 Link failure between clusters

In this case, there is no effect upon the topology in the cluster. Node *u* and node *v* only have to check their *gw*() status. If there is no change, there is nothing to do. Otherwise, nodes *u* and *v* must update their cluster information and notify their cluster head. Fig. 8 illustrates the link failure between clusters.

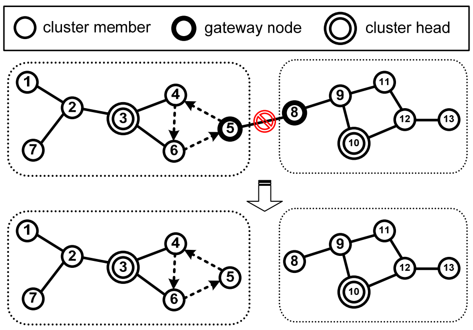


Fig. 8. An example of link failure between clusters.

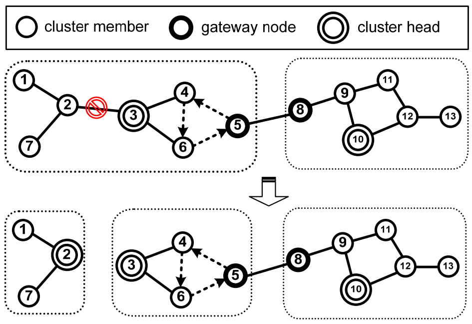


Fig. 9. An example of link change within a cluster.

2.3.5 Link failure in a cluster

If nodes *u* and *v* belong to a same cluster, then, the topology of the cluster would be changed when the link between *u* and *v* breaks. In this case, nodes *u* and *v* have to notify their cluster head about this change. After receiving the message, cluster head will update its link state and check if it can still dominate node *u* and *v*. If true, it does nothing, and nodes u and v have to update their cluster information. Otherwise, the node that can't be managed by the original cluster head has to find a new cluster in its neighbors and join it, or itself becomes a cluster head. Fig. 9 is an example of link change in the cluster. While link failure is occurred between node 2 and node 3, node 2 can't find any new cluster to join, thus it turns itself into a new cluster head (after competing with node 1 and node 7).

2.3.6 Cluster head retirement

Most of the existing clustering algorithms have maintenance schemes to deal with topology change. However, the number of clusters in the network will increase constantly while maintaining. Some of them suggest that the whole network should restart clustering after a period of time, but this significantly affects the network performance and is not a practical solution for a distributed network system. Hence, we propose a retirement scheme for cluster heads that can enforce some unsuitable cluster heads to become normal nodes and then join their neighbor clusters. The performance of the network can be kept without re-clustering of the whole network.

In our retirement scheme, while a cluster head p recognizes:

1. $\exists y \in N_1(p)$ and $dom(y) \neq p$ and $hth(y) < 2$.
2. $ncm(y) > ncm(p)$.
3. There is no pendant node in cluster p.

Then, the cluster head p will retire and join one of the neighbor clusters. Fig. 10 is an example of the cluster head retirement. We do not allow a cluster head to retire while it manages pendant nodes, since once it retires; the pendant nodes may not easily to find a neighbor cluster to join and thus results in forming new clusters. This may constantly incur unnecessary role changes and cluster switches.

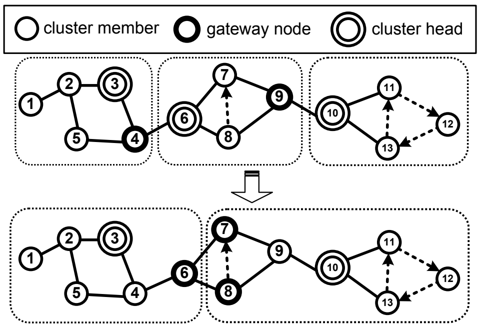


Fig. 10. An example of the retirement scheme.

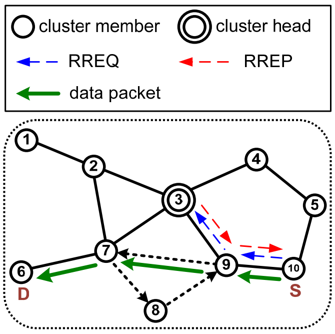


Fig. 11. Intra-cluster routing.

3. ROUTING PROTOCOL AND ROUTE MAINTENANCE

In this section, we present a hybrid routing protocol which consists of two parts: intra-cluster routing and inter-cluster routing. We employ the idea of the proactive (table-driven) scheme in intra-cluster routing, and the reactive (on-demand) scheme is used in inter-cluster routing. Our routing protocol is an improvement of ZRP [14] and CBRP [17]. In ZRP, every node in the network has to maintain *k*-hop information which overlaps among neighbor nodes. CBRP also requires each node to maintain 2-hop link states for source routing, route shortening, and local repair. In our scheme, intra-cluster routing information is maintained by cluster heads, each node only maintains its 1-hop neighbor information. The route discovery procedure of inter-cluster routing is based on communication between cluster heads and gateway nodes. Each of them is described in detail in the following sections.

3.1 Intra-Cluster Routing

After clustering, each cluster head maintains the link states of its cluster. If a routing session starts, the source node *u* first sends a route request packet (RREQ) to its cluster head *dom*(*u*), if both the source node *u* and the destination node *v* belong to the same cluster (*dom*(*u*) = *dom*(*v*)), then *dom*(*u*) will reply node *u* a shortest route from node *u* to node *v* based on the link states maintained by the cluster head. Thus, the data packets of source node *u* can be transmitted to the destination node *v* hop by hop along this route. Note that the replied route has been shortened and needs not always pass through the cluster head. For example, in Fig. 11, source node 10 sends a RREQ to its cluster head node 3, node 3 then replies node 10 a shortest route 10-9-7-6. Since the link management has been applied on each node, unidirectional links can be used for both routing and control packet transmitting, which enriches the connectivity of the network and minimizes the path length.

3.2 Inter-Cluster Routing

When the source and the destination are in different clusters, the inter-cluster routing is responsible for route construction. Once the cluster head can't find the destination in its cluster member list, it initiates the inter-cluster route construction. In inter-cluster routing, the cluster head of the source node sends a RREQ to the network relying on its gateway nodes. The RREQ of the inter-cluster routing is transmitted by <cluster head - gateway nodes - cluster head>. When a cluster head wants to send a message to its adjacent cluster heads, it sends the message to its gateway nodes, and then the gateway nodes can forward the message to the adjacent cluster heads. We explain the inter-cluster route construction with an example. Assuming the source is node *u* and the destination is node *v*. The inter-cluster route construction between node *u* and node *v* is to construct a route from *dom*(*u*) to *dom*(*v*). First, the cluster head of node *u* broadcasts a RREQ to the gateway nodes of its cluster. To avoid redundant packets flooding, a sequence number is added to the RREQ and each node only forwards the RREQ once with the same sequence number. The cluster head of destination node *v* will soon receive the RREQ from *dom*(*u*), and then it sends a reply packet along the route traveled by the RREQ. When each cluster head on this route receives the route reply packet, it shortens the route by its own link state information (the same as intra-cluster routing) and then continues the route reply. Thus, node *u* finally obtains a route to node *v*. In Fig. 12, node 3 first sends the message to its gateway node 5, node 5 then sends the message to its gateway neighbor node 11. Node 11 then sends the message to its cluster head node 12. In this way, the route can be

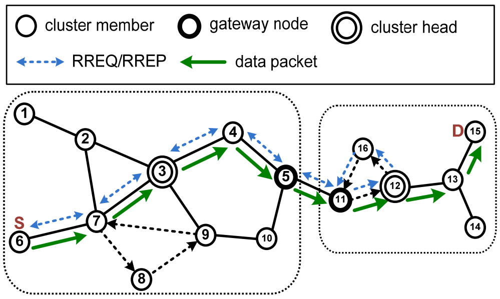


Fig. 12. Inter-cluster routing.

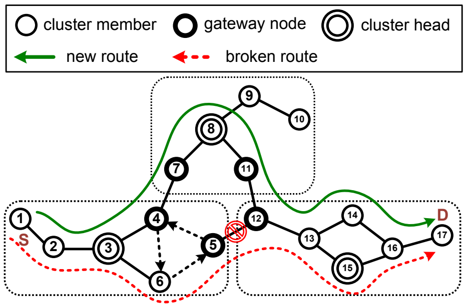


Fig. 13. An example of route maintenance.

constructed among clusters. In fact, in many other routing protocols, the route from node 6 to node 15 doesn't exist since the links between nodes 11, 12, and 16 are unidirectional.

3.3 Route Maintenance

The constructed route may disconnect due to the mobility of nodes, an efficient route maintenance mechanism is essential to ensure the success of packet delivery. For example, in Fig. 13, the route 1-2-3-6-5-12-13-15-16-17 is established from node 1 (source) to node 17 (destination). While node 5 realizes the broken link between nodes 5 and node 12, it notifies its cluster head to perform the local repair, that is to find a route to reconnect the next cluster of the original route. If the cluster has gateway nodes to reach the next cluster of the original route, the local route repair is done by forwarding the packet to the gateway node which connects to the next cluster. Once the cluster can find a gateway node to reconnect the next cluster, a local RREQ is issued. The local RREQ only transmits to the adjacent clusters, if the route could be reconnected, the overhead of new route reconstruction is prevented. If the original route cannot be reconnected by adjacent clusters, a RERR (route error) is transmitted back to the source node and a new global RREQ will be issued for new route construction. In this example, the route is reconnected by adjacent cluster and the route is locally repaired (1-2-3-4-7-8-11-12-13-14-16-17).

In a medium to large size ad hoc network, local repair is helpful to reduce route reconstruction overhead and also decreases the packet delay. Compare our route maintenance mechanism with CBRP [17], CBRP uses source routing after a route is constructed, local repair will be activated more frequently owing to the nature of source routing, and the local repair range is limited to 2-hop neighbors maintained by each node. In our scheme, since the intra-cluster route is maintained periodically by the cluster head, local repair is only initiated while link failure occurs between clusters, which is more effective and efficient.

4. PERFORMANCE EVALUATION

In the first part of this section, the performance of different clustering algorithms are evaluated, since in a cluster-based routing scheme, the stability of a clustering algorithm has a great effect on routing performance and maintenance overhead. In addition to evaluate the performance of different weight function considerations, we also compare our clustering algorithm with 2-hop SK-clustering [41], and 2-hop Lowest-ID clustering with Least Cluster Change (LCC) [7] by metrics such as the number of clusters and the performance of stability. The clustering algorithm of CBRP [17] is very similar to Lowest-ID with LCC, the only difference between them is mainly on routing schemes. In the second part, the proposed routing protocol is compared with DSR [18] and ZRP [14]. The size of simulation area includes $1000m \times 1000m$, $2000m \times 2000m$, $2200m \times 600m$. The number of nodes is from 100 to 500 and the transmission range (TR) varies from $100m$ to $250m$, simulation time is 300 seconds, mobility speed is from $0m/s$ to $30m/s$ with random waypoint mobility model. We evaluate the performance under unidirectional network environments by controlling the percentage of nodes to have a smaller transmission range. A percentage of nodes are set to have a smaller transmission range (*i.e.* 30% nodes have 50% of the original transmission range) to create unidirectional links in the network.

4.1 Analysis of Cluster Performance

We consider the following parameters for the performance analysis of clustering algorithms: (1) Number of clusters: in the cluster-based routing protocol, the overhead mainly comes from the clustering procedure and the communication among clusters. Thus, the fewer number of clusters is formed by a clustering algorithm, the lower overhead is consumed; (2) Number of role changes: the role change is calculated by summing up the status changes of all nodes through the whole simulation (*i.e.* the status of a node changes from a cluster head to a member node, or from a member node to a cluster head). This parameter is a measurement of cluster stability, the smaller number of role changes means better stability and lower maintenance overhead of the clustering algorithm. (3) Number of cluster switches: the cluster switch is a member node changes its cluster head from one to the other. This parameter is also used for measuring the stability of a clustering algorithm. A stable clustering algorithm makes good use of the network resources since it handles the changing topology better thus incurring less overhead. Therefore, we believe that a good clustering algorithm should avoid frequently role changes and cluster switches as well.

First, we show the cluster performance of different weight functions in Fig. 14. The detail of clustering algorithms and environment settings are described in Tables 3 and 4. We vary the coefficients of weight function defined in Eq. (3) of section 2.2, to see the number of clusters formed by different weight functions. The High-Connectivity Clustering (HCC) [15], Lowest-ID clustering (LID) [15], and SK-clustering (SK) [41] are also compared. Fig. 14 shows that both HCC and LID clustering algorithms generate more clusters than the others. Especially is the LID, since it does not use any useful information of the network for clustering. The proposed weight based clustering algorithm outperforms all the other clustering algorithms. Though the performance differences are close among different weight functions, the W_{our} generates least clusters as compares with all the other weight functions in all simulation environments. This result validates the performance of the proposed weight function, and as mentioned above, the less clusters formed implies the less control and maintenance overhead produced.

Table 3. Detail of weight function in each clustering algorithm in Fig. 14.

Abbreviation	Descriptions
W_{our}	$w(p) = (0.7\times s(p) + 0.2\times t(p) + 0.1\times deg(p))$ As proposed in section 2.2, Eq. (3).
W_A	$w(p) = (0.5\times s(p) + 0.4\times t(p) + 0.1\times deg(p))$
W_B	$w(p) = (0.8\times s(p) + 0.1\times t(p) + 0.1\times deg(p))$
W_C	$w(p) = (0.3\times s(p) + 0.5\times t(p) + 0.2\times deg(p))$
W_D	$w(p) = (0.1\times s(p) + 0.8\times t(p) + 0.1\times deg(p))$
HCC	The high-connectivity clustering algorithm [15].
LID	The Lowest-ID clustering algorithm [15].
SK	The SK-clustering algorithm [41].

Table 4. Detail of various simulation environments in Fig. 14.

Abbreviation	Simulation environment details
E1	1000*m* × 1000*m*, 100 nodes, TR = 200*m*
E2	1000*m* × 1000*m*, 200 nodes, TR = 200*m*
E3	2000*m* × 2000*m*, 100 nodes, TR = 200*m*
E4	2000*m* × 2000*m*, 200 nodes, TR = 200*m*
E5	2000*m* × 2000*m*, 100 nodes, TR = 250*m*
E6	2200*m* × 600*m*, 100 nodes, TR = 200*m*
E7	2200*m* × 600*m*, 200 nodes, TR = 200*m*
E8	2200*m* × 600*m*, 100 nodes, TR = 250*m*

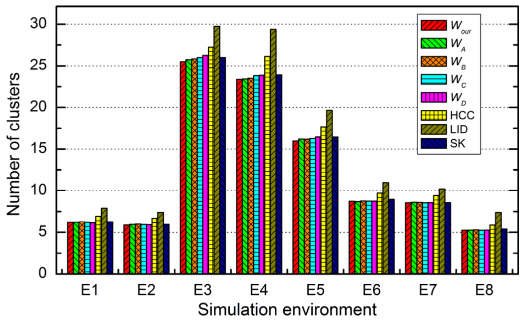


Fig. 14. Number of clusters of different clustering algorithm vs. simulation environment.

Figs. 15 and 16 show the number of clusters with unidirectional links in the network. Our scheme provides better performance in the number of clusters, especially when the percentage of unidirectional links grows. Figs. 17 to 19 show that even in the network environment with only bidirectional links, our scheme still outperforms the others in the number of clusters. Figs. 18 and 19 are role changes and cluster switches, both of them indicate the stability of our clustering algorithm. Note that LCC pays a lot of maintenance overhead to keep the number of clusters low. On the other hand, our scheme performs well in both role changes and in cluster switches.

As expected, in Figs. 20 and 21, higher mobility causes higher role changes and cluster switches. The results indicate that our scheme provide better stability while facing

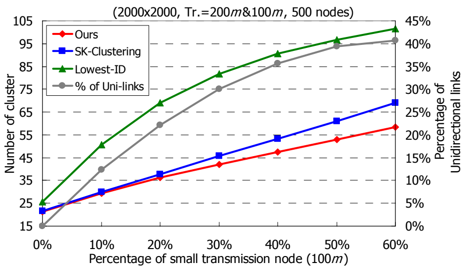


Fig. 15. Percentage of small transmission node vs. number of clusters and percentage of unidirectional links.

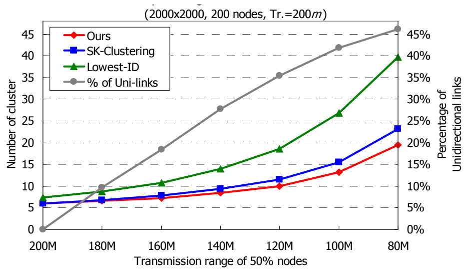


Fig. 16. Transmission range of 50% nodes vs. number of clusters and percentage of unidirectional link.

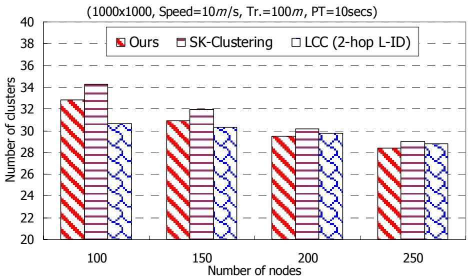


Fig. 17. Average number of clusters vs. number of nodes.

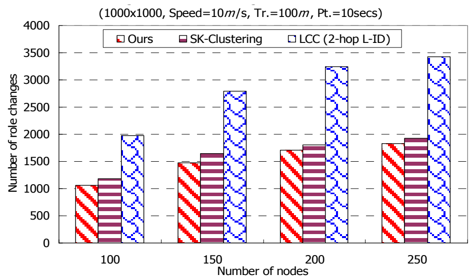


Fig. 18. Average number of role changes vs. number of nodes.

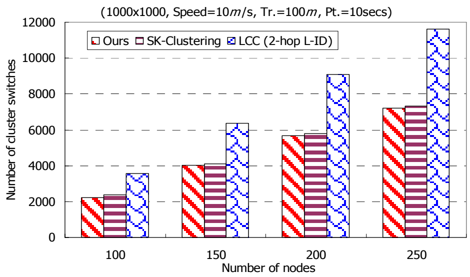


Fig. 19. Average number of cluster switches vs. number of nodes.

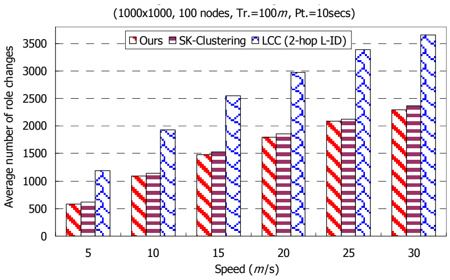


Fig. 20. Average number of role changes vs. speed.

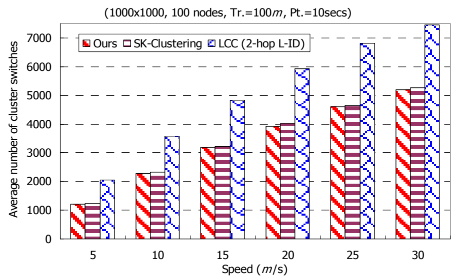


Fig. 21. Average number of cluster switches vs. speed.

dynamic topology. In this section, the results show that the proposed weight function is effective in generating clusters which are more stable, and thus less maintenance overhead is required.

4.2 Analysis of Routing Performance

In the course of this simulation, the following parameters are measured: (1) Delivery ratio: the ratio of data packets delivered to the destination to those generated by the source. (2) Number of RREQ forwarding nodes: when a routing session is started, the number of nodes in the network that helps re-broadcasts the RREQ packet issued by the source. (3) Average hop count: the average hop count from the source to the destination of the route. Figs. 22 to 24 show the delivery ratio of our cluster based routing protocol (CRP), comparing with ZRP and DSR. Because the maintenance scheme of the proposed cluster routing protocol is mainly based on the cluster information update, its delivery ratio drops in high mobility network environment. Once the mobility is low, the performance of CRP is similar to the others.

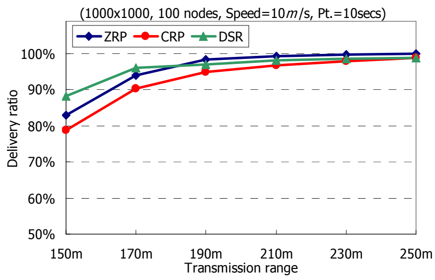


Fig. 22. Delivery ratio vs. transmission range.

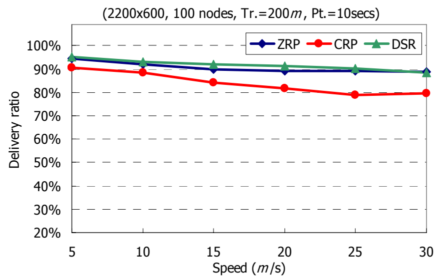


Fig. 23. Delivery ratio vs. speed.

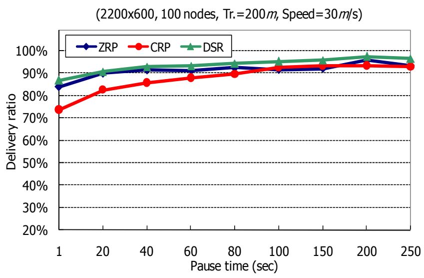


Fig. 24. Delivery ratio vs. Pause time.

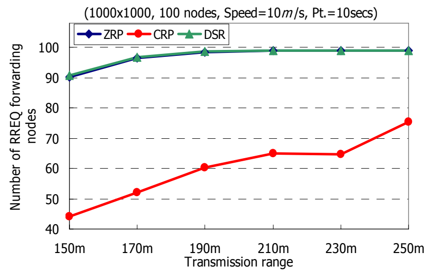


Fig. 25. Number of RREQ forwarding nodes vs. transmission range.

In most of on-demand routing protocols such as DSR, the source has to broadcast a route request packet (RREQ) to the entire network for discovering a route. The blind flooding of RREQ packets causes the waste of bandwidth and the congestion of the network. In Figs. 25 to 27, our protocol does not flood the RREQ into the whole network thus avoiding large number of RREQ forwarding nodes. Although ZRP uses bordercasting to prevent flooding, when the network connectivity is rich, the RREQ is still forwarded to almost the entire network.

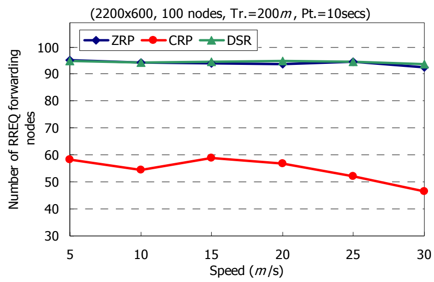


Fig. 26. Number of RREQ forwarding nodes vs. speed.

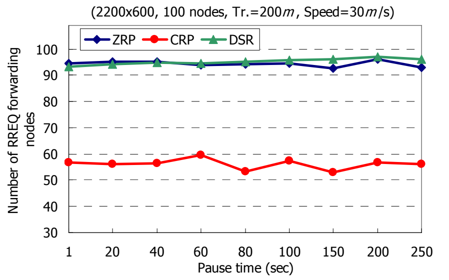


Fig. 27. Number of RREQ forwarding nodes vs. pause time.

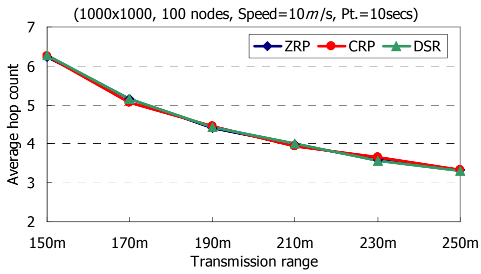


Fig. 28. Average hop count vs. transmission range.

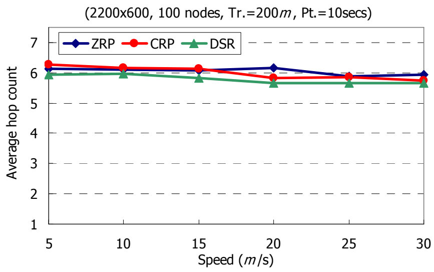


Fig. 29. Average hop count vs. speed.

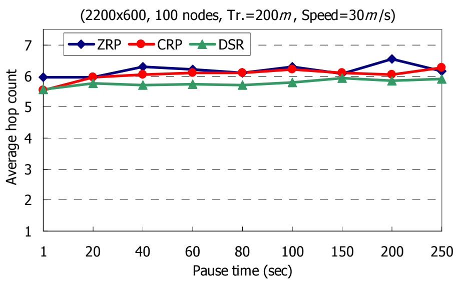


Fig. 30. Average hop count vs. pause time.

The average hop count is shown in Figs. 28 to 30. As expected, the three routing protocols construct paths with similar hop distances. The path constructed by DSR is the shortest since it uses shortest hop distance as the metric of route construction. Although our routing protocol is a cluster based routing, which still achieves a short average hop count compares with DSR and ZRP.

5. CONCLUSIONS

In this paper, we propose an efficient cluster-based routing protocol that supports unidirectional network environments, where the connectivity is enriched by managing and utilizing unidirectional links. The maintenance of both clustering and routing are also discussed to adapt to topology change. The proposed protocol is suitable for medium to large networks with low to medium mobility. The simulation results show that our scheme has better performances, less number of role changes and cluster switches, which provides better stability in mobile network environment.

REFERENCES

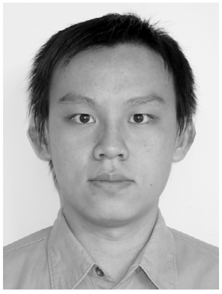
1. M. Abolhasan, J. Lipman, and J. Chicharo, "A routing strategy for heterogeneous mobile ad hoc networks," in *Proceedings of the 6th IEEE Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication*, Vol. 1, 2004, pp. 13-16.
2. A. D. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Proceedings of the 3rd IEEE Symposium on Application-Specific Systems and Software Engineering Technology*, 2000, pp. 25-32.
3. A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," in *Proceedings of the 19th IEEE Annual Joint Conference of the IEEE Computer and Communications Societies* (*INFOCOM*), Vol. 1, 2000, pp. 32-41.
4. T. Asano, H. Unoki, and H. Higaki, "LBSR: routing protocol for MANETs with unidirectional links," in *Proceedings of the 18th International Conference on Advanced Information Networking and Applications*, Vol. 1, 2004, pp. 219-224.
5. M. Chatterjee, S. K. Sas, and D. Turgut, "An on-demand weighted clustering algorithm (WCA) for ad hoc networks," in *Proceedings of the IEEE Global Telecommunications Conference*, Vol. 3, 2000, pp. 1697-1701.
6. Y. S. Chen, Y. C. Tseng, J. P. Sheu, and P. H. Kuo, "An on-demand, link-state, multi-path QoS routing in a wireless mobile ad hoc network," *Computer Communications*, Vol. 27, 2004, pp. 27-40.
7. C. C. Chiang, H. K. Wu, W. Liu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks with fading channel," in *Proceedings of the IEEE Singapore International Conference on Networks*, 1997, pp. 197-211.
8. W. Choi and M. Woo, "A distributed weighted clustering algorithm for mobile ad hoc networks," in *Proceedings of the International Conference on Internet and Web Applications and Services/Advanced International Conference on Telecommunications*, 2006, pp. 73-78.
9. J. A. Cobb, "Forward-only uni-directional routing," in *Proceedings of the 11th International Conference on Computer Communications and Networks*, 2002, pp. 370-375.
10. S. K. Dhurandher and G. V. Singh, "Weight based adaptive clustering in wireless ad hoc networks," in *Proceedings of the IEEE International Conference on Personal Wireless Communications*, 2005, pp. 95-100.
11. Y. X. Feng, L. L. Zhao, and G. X. Wang, "A clustering algorithm applied to the network management on mobile ad hoc network," in *Proceedings of the International Conferences on Info-tech and Info-net*, Vol. 2, 2001, pp. 626-631.
12. M. Gerla, Y. Z. Lee, J. S. Park, and Y. Yi, "On demand multicast routing with unidirectional links," in *Proceedings of the IEEE Wireless Communications and Networking Conference*, Vol. 4, 2005, pp. 2162-2167.

13. Z. J. Haas and M. R. Pearlman, "The performance of query control schemes for the zone routing protocol," *IEEE/ACM Transactions on Networking*, Vol. 9, 2001, pp. 427-438.
14. Z. J. Haas, M. R. Pearlman, and P. Samar, "The zone routing protocol (ZRP) for ad hoc networks," Internet Draft, draft-ietf-manet-zone-zrp-04.txt, IETF MANET, 2002.
15. T. C. Hou and T. J. Tsai, "Adaptive clustering in a hierarchical ad hoc network," in *Proceedings of the International Computer Symposium Workshop on Computer Networks*, *Internet*, *and Multimedia*, 1998, pp. 171-176,
16. G. Jakllari, W. Luo, and S. V. Krishnamurthy, "An integrated neighbor discovery and MAC protocol for ad hoc networks using directional antennas," *IEEE Transactions on Wireless Communications*, Vol. 6, 2007, pp. 1114-1124.
17. M. Jiang, J. Li, and Y. C. Tay, "Cluster based routing protocol (CBRP)," Internet Draft, draft-ietf-manet-cbrp-spec-01.txt, IETF MANET, 1999.
18. D. B. Johnson, D. A. Maltz, and Y. C. Hu, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," Internet Draft, draft-ietf-manet-dsr-10.txt, IETF MANET, 2004
19. T. J. Kwon, M. Gerla, V. K. Varma, M. Barton, and T. R. Hsing, "Efficient flooding with passive clustering-an overhead-free selective forward mechanism for ad hoc/sensor networks," in *Proceedings of the IEEE*, Vol. 91, 2003, pp. 1210-1220.
20. B. Liang and Z. J. Haas, "Hybrid routing in ad hoc networks with a dynamic virtual backbone," *IEEE Transactions on Wireless Communications*, Vol. 5, 2006, pp. 1392-1405.
21. H. Mahmoud and A. Fahmy, "Power-aware dominating set calculations in mobile ad hoc wireless networks with unidirectional links," in *Proceedings of the 10th IEEE International Conference on Electronics*, *Circuits and Systems*, Vol. 3, 2003, pp. 1272-1275.
22. D. A. Maltz, J. Broch, J. Jetcheva, and D. B. Johnson, "The effects of on-demand behavior in routing protocols for multihop wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, 1999, pp. 1439-1453.
23. A. B. McDonald and T. F. Znati, "A mobility-based framework for adaptive clustering in wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, 1999, pp. 1466-1487.
24. H. Morino, T. Miyoshi, and M. Ogawa, "Ad hoc unidirectional routing protocol based on relay control of route requests," in *Proceedings of the 7th International Symposium on Autonomous Decentralized Systems*, 2005, pp. 675-680.
25. H. Morino, T. Miyoshi, and M. Ogawa, "Unidirectional ad hoc routing with efficient route reconstruction using relay control of route requests," in *Proceedings of the 61st IEEE Vehicular Technology Conference*, Vol. 4, 2005, pp. 2504-2508.
26. S. Nesargi and R. Prakash, "A tunneling approach to routing with unidirectional links in mobile ad hoc networks," in *Proceedings of the 9th International Conference on Computer Communications and Networks*, 2000, pp. 522-527.
27. M. R. Pearlman and Z. J. Haas, "Determining the optimal configuration for the zone routing protocol," *IEEE Journal on Selected Areas in Communications*, Vol. 17, 1999, pp. 1395-1414.
28. M. R. Pearlman, Z. J. Haas, and B. P. Manvell, "Using multi-hop acknowledgements to discover and reliably communicate over unidirectional links in ad hoc networks,"

in *Proceedings of the IEEE Wireless Communications and Networking Conference*, Vol. 2, 2000, pp. 532-537.

29. C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," RFC 3561, IETF Network Working Group, 2003.
30. C. E. Perkins, E. M. Royer, S. R. Das, and M. K. Marina, "Performance comparison of two on-demand routing protocols for ad hoc networks," *IEEE Personal Communications*, Vol. 8, 2001, pp. 16-28.
31. Y. Qin, "Analysis of cluster-based hierarchical routing in ad hoc wireless networks," *Electronics Letters*, Vol. 42, 2006, pp. 474-476.
32. B. Ramachandran and S. Shanmugavel, "Performance analysis of cross layer AODV for heterogeneously powered ad hoc networks," in *Proceedings of the IFIP International Conference on Wireless and Optical Communications Networks*, 2006, pp. 1-6.
33. R. Ramanujan, S. Takkella, J. Bonney, and K. Thurber, "Source-initiated adaptive routing algorithm (SARA) for autonomous wireless local area networks," in *Proceedings of the 23rd Annual Conference on Local Computer Networks*, 1998, pp. 109-118.
34. V. Ramasubramanian, R. Chandra, and D. Mosse, "Providing a bidirectional abstraction for unidirectional ad hoc networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 3, 2002, pp. 1258-1267.
35. V. Ramasubramanian and D. Mosse, "Statistical analysis of connectivity in unidirectional ad hoc networks," in *Proceedings of the International Conference on Parallel Processing Workshops*, 2002, pp. 109-115.
36. E. M. Royer and C. K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *IEEE Personal Communications*, Vol. 6, 1999, pp. 46-55.
37. P. Samar, M. R. Pearlman, and Z. J. Haas, "Independent zone routing: an adaptive hybrid routing framework for ad hoc wireless networks," *IEEE/ACM Transactions on Networking*, Vol. 12, 2004, pp. 595-608.
38. V. Shah and S. Krishnamurthy, "Handling asymmetry in power heterogeneous ad hoc networks: a cross layer approach," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005, pp. 749-759.
39. A. A. Siddiqui and R. Prakash, "Effect of availability factor threshold and clustering gap on performance of clustering mechanisms for multi-cluster mobile ad hoc networks," in *Proceedings of the IEEE International Conference on Communications*, Vol. 5, 2002, pp. 3296-3300.
40. P. Sinha, S. V. Krishnamurthy, and S. Dao, "Scalable unidirectional routing with zone routing protocol (ZRP) extensions for mobile ad hoc networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference*, Vol. 3, 2000, pp. 1329-339.
41. D. C. Su, S. F. Hwang, C. R. Dow, and Y. W. Wang, "An efficient K-hop clustering routing scheme for ad hoc wireless networks," *Journal of Internet Technology*, Vol. 3, 2001, pp. 139-146.
42. Y. Y. Su, S. F. Hwang, and C. R. Dow, "An efficient multi-source multicast routing protocol in mobile ad hoc networks," in *Proceedings of the 11th International Conference on Parallel and Distributed Systems*, Vol. 1, 2005, pp. 8-14.
43. N. Vanitchanant and J. Silvester, "Unidirectionality in ad hoc networks: a simulation

study," in *Proceedings of the IEEE Military Communications Conference*, Vol. 2, 2003, pp. 1298-1304.
44. N. Vanitchanant, J. Silvester, and V. Vipanunt, "Optimization to multicasting protocol in unidirectional ad hoc networks," in *Proceedings of the IEEE International Conference on Wireless And Mobile Computing*, *Networking and Communications*, Vol. 3, 2005, pp. 329-335
45. J. Wu, "Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, 2002, pp. 866-881.



Yi-Yu Su (蘇奕宇) was born in 1977. He received the B.S. degrees in Computer and Information Science from Aletheia University, Taiwan, in 2000, and received the M.S. degree in Information Engineering from Feng Chia University, Taiwan, in 2002. He is currently a candidate for the Ph.D. degree in the Department of Information Engineering, Feng Chia University, Taichung, Taiwan. His research interests include mobile computing, mobile ad hoc networks, and wireless sensor networks.

Shiow-Fen Hwang (黃秀芬) was born in 1963. She received her B.S., M.S. and Ph.D. degrees in Applied Mathematics from National Chiao Tung University, Taiwan, in 1985, 1987 and 1991, respectively. Currently, she is an Associate Professor in the Department of Information Engineering, Feng Chia University, Taiwan. Her research interests include interconnection networks, mobile computing, and computer algorithms.

Chyi-Ren Dow (竇其仁) was born in 1962. He received the B.S. and M.S. degrees in Information Engineering from National Chiao Tung University, Taiwan, in 1984 and 1988, respectively, and the M.S. and Ph.D. degrees in Computer Science from the University of Pittsburgh, USA, in 1992 and 1994, respectively. Currently, he is a Professor in the Department of Information Engineering, Feng Chia University, Taiwan. His research interests include mobile computing, ad hoc wireless networks, agent techniques, fault tolerance, and learning technology.