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
Feng Chia Unverisity, Taichung, Taiwan 407, R.O.C.
+Email: {sys1094@ms9.hinet.net}
*Email: {sfhwang, crdow@fcu.edu.tw}
TEL: 886-4-24517250 Ext. 3723 FAX: 886-4-24516101

***Abstract*-Wireless ad hoc networks are dynamically organized by a collection of wireless mobile hosts. In ad hoc networks, any pre-established wired or wireless infrastructures and the centralized administration, such as the base stations, are unnecessary. The mobile host in ad hoc network can move arbitrarily thus the network topology changes dynamically and no centralize management mechanism can be used. Due to the properties of radio communication in wireless network, there exists unidirectional links between mobile hosts thus result in the difficulty of both link utilization and routing. In this paper, we take the advantages of the multi-hop acknowledgement and employ the clustering technique to design an efficient hybrid routing algorithm in ad hoc network with unidirectional links. Simulation results demonstrate the stability and efficiency of the proposed algorithm.**

Keywords: ad hoc network, unidirectional link, clustering, mobile host, on-demand (reactive), routing, table-driven (proactive).

I. Introduction

An ad hoc network is a collection of wireless mobile hosts forming a temporary wireless network. The communication between mobile hosts can be accomplished via the nearby mobile hosts interchanging messages. In ad hoc networks, any pre-established wired or wireless infrastructure and the centralized administration, such as the base stations, are unnecessary. The mobile host can move arbitrarily and no centralized management mechanisms can be used. Thus, the ad hoc network is specially important and useful in battlefield or disaster area. Under the limited resources such as network bandwidth, memory capacity, and battery power, the efficiency of routing schemes in ad hoc wireless networks become more important and challenging. In ad hoc networks, current routing protocols can be classified as proactive(table-driven) or reactive(on-demand) [2,3,15]. The proactive routing schemes obtain route information from each node’s routing table quickly by periodically updating topology information, but needs to update frequently and keep large size of routing table. The reactive routing schemes find and maintain routes only when it is used for communication. However, it suffers the delay and flooding overhead of route request. To take the advantages of these two schemes, clustering technique was proposed by many researchers [1,4,6,7,10,11,17]. The mobile hosts in the network can be divided into several clusters. In each cluster, a mobile host will be elected as a cluster head to manage the cluster members and exchange important information. By using of the clustering technique, a large size of network can be divided into several sub-networks and only the cluster head maintains the local topology and routing information about its own cluster. Thus, the overhead of topology maintenance can be reduced and network-wide flooding can be prevented.

There are many clustering algorithms, such as the Lowest-ID Clustering [6], High-Connectivity Clustering [6], Label-Clustering algorithm [10], Weighted Clustering algorithm [4] and so on[5]. These presented clustering algorithms usually result in lots cluster heads and poor stability that reduce the advantages of the cluster structure. Most of current routing protocols and clustering algorithms assume that all nodes in the network have the same transmission range thus all links are bidirectional in the network. In fact, this does not reflect the real scenario where radio transmission range may drop caused by power consumption of the mobile hosts. Due to the characteristics of the radio transmission in wireless network, there exists unidirectional links between mobile hosts caused by differences in transmission range, hidden terminal problem or radio interference [8,9,14,16,18,19,20]. Undiscovered unidirectional links represent untapped network capacity and reduced network connectivity. In this paper, we take the advantages of the multi-hop acknowledgement and employ the clustering technique to design an efficient hybrid routing algorithm in ad hoc network with unidirectional links. We also propose the maintenance scheme that ensures our clustering and routing are still efficient even if the network topology changes.

The rest of this paper is organized as follows. Our clustering and routing algorithm, maintenance schemes are described in section II. The simulation results are presented in section III. Conclusions are offered in section IV.

II. Clustering and Routing Algorithms

In this section, we introduce our clustering and routing algorithm. First, we discuss the link management in our scheme, than we propose our clustering algorithm. After clustering, a routing algorithm is presented to achieve the communication between mobile hosts. Finally, the maintenance mechanism is discussed which adapt our algorithm to the changing topology caused by node mobility.

2.1 Link Management of Unidirectional Links

In ad hoc networks with only bidirectional links, the mobile hosts (nodes) use Hello and I-Hear-You to detect nodes within the transmission range. If the transmission range of the nodes were different, the networks may exist unidirectional links thus the traditional link-layer acknowledgement (ACK, Hello and I-Hear-You) may not able to detect the unidirectional links properly (Fig. 1).

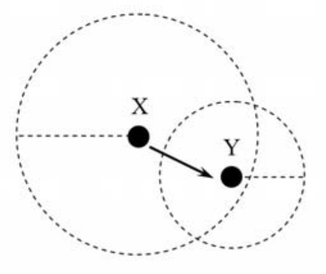

Fig. 1. Transmission range induced unidirectionality.

Therefore, we employ the multi-hop acknowledgement [13] and tunneling [12] techniques to detect unidirectional links between nodes. First, we define the Time-To-Live (TTL) of the broadcast acknowledgement packets which make the acknowledgement can be re-broadcasted by the neighbor nodes and detect the unidirectional links. In figure 2, node A first broadcasts Hello packet to its neighbors, but node C can not reply ACK to node A directly because of the unidirectional link from node A to C, thus, node A may lose its neighbor C in this way. Now, by using the multi-hop acknowledgement and tunneling approaches, node C's ACK may be transmitted back to node A via node D or B and the unidirectional link from node A to C can be established. Then, node A sends a packet to node C which including the path C-D-A or C-B-A. Therefore, node C also learns a way to node A such as C-D-A or C-B-A. By this approach, node A and node C can learn the existence of each other and the link (route) between node A and C can be learned. As a result, the unidirectional links in the network can be detected and the link management can be accomplished.

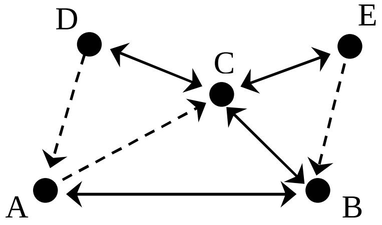

Fig. 2. Multi-hop link layer acknowledgements.

2.2 Clustering Algorithm

Initially, we define the *cluster information* that is maintained on each node. The *cluster information* is very important that containing the necessary information in our clustering algorithm, it will be described in the following section. After collecting the *cluster information*, the node can determine its own status easily by exchanging *cluster information* with its neighbors. The *cluster information* is also used for maintenance and routing. Before we introduce the clustering algorithm, some notations must be defined:

V	*the set of all nodes.*
$N_1(v)$	*the set of 1-hop neighbors of v.*
$N_2(v)$	*the set of 2-hop neighbors of v.*
$d(u,v)$	*the least hops from node u to node v.*
label(v)=U	*indicates the status of node v is undetermined.*
label(v)=D	*indicates the status of node v is a member of a cluster.*
label(v)=H	*indicates the status of node v is a cluster head of a cluster.*
pendant node	*a node has only one neighbor which is bidirectional linked.*
$s(v) = \lvert\{x \mid x \text{ is a pendant node} \;\&\; x \in N_2(v)\}\rvert$	*the number of pendant nodes in v's 2-hop neighbors*
$t(v) = \lvert\{x \mid label(x) = U \;\&\; x \in N_2(v)\}\rvert$	*the number of nodes which status is U in v's 2-hop neighbors*
Neighbor table	*including 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 route backs to each unidirectional neighbor is additionally maintained in unidirectional neighbor table (by broadcasting multi-hop acknowledgement).*

Cluster Information(id(v), dom(v), deg(v), hth(v), next(v), ncm(v), gw(v))

id(v)	*id of node v.*
dom(v)	*cluster head of node v.*
deg(v)	*degree of node v.*
hth(v)	*minimum hops to cluster head of node v*
next(v)	*the next hop node's id to the cluster head of node v*
ncm(v)	*number of the current cluster members*
gw(v)	*TRUE if node v is a gateway node of its cluster, FALSE, otherwise(Assuming that there is a node called u which is one of the node v's neighbors. If cluster head of*

node u is not the same with node v, both node u and v are gateway nodes of their own cluster respectively).

In ours algorithm, two assumptions should be declared:

1. Nodes can transmit and receive packets correctly in limited time.
2. Nodes can detect the link states in their transmission ranges in limited time.

Before executing the clustering algorithms, a node with status "U" first exchanges messages within its 2-hop neighbors to build the neighbor tables and collects the *cluster information*. Then, each node *p* will execute the following procedure:

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

For each cluster head, the following procedure will be executed:

```
become_head(x)
{
    label(x)=H;
    hth(x)=0;
    $\forall\ y \in \{p \mid p \in N_1(x)\ \&\ label(p) = U\}$

    {
        label(y)=D;
        hth(y)=1;
        dom(y)=x;
        next(y)=x;

    $\forall\ z \in \{q \mid q \in N_1(y)\ \&\ label(q) = U\}$
        {
            label(z)=D;
            hth(z)=2;
            dom(z)=x;
            next(z)=y;
        }
    }
}
```

In our clustering algorithm, the radius of each cluster is 2, and we ensure that each member node (status is D) belongs to one cluster head which is one of its 2-hop neighbor nodes. As above, there is a weight function $w(p)$ in the clustering algorithm for nodes to calculate and compare their weight with the neighbors to determine the status of each other. For each node p, the weight function $w(p)$ is defined as:

$$w(p) = (0.7 \times s(p) + 0.2 \times t(p) + 0.1 \times deg(p))$$

$s(p)$ is the number of pendant nodes in p's 2-hop neighbors and $t(p)$ is the number of nodes which status is U in p's 2-hop neighbors, the $deg(p)$ is the degree of node p. One thing must be noticed is that we also define the degree of a bidirectional link equals to 1 and a unidirectional link equals to 0.5. For example, in figure 3, the weight of node 1, node 3 and node 7 are:

$$w(1) = (0.7 \times 1 + 0.2 \times 4 + 0.1 \times 1) = 0.7 + 0.8 + 0.1 = 1.6$$

$$w(3) = (0.7 \times 2 + 0.2 \times 10 + 0.1 \times 4) = 1.4 + 2.0 + 0.4 = 3.8$$

$$w(7) = (0.7 \times 2 + 0.2 \times 9 + 0.1 \times 5) = 1.4 + 1.8 + 0.5 = 3.7$$

The weight of the other nodes in the network can be calculated in this way.

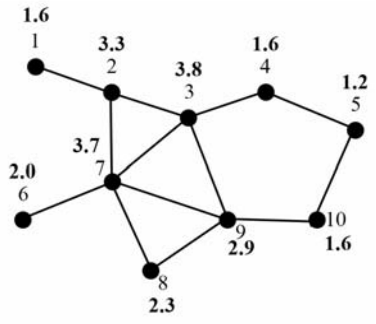

Fig. 3. Weight of nodes with status U.

After the weight function execution, each node needs to compare the weight with neighboring nodes for status determination. We use two times neighbor comparison mechanism to complete the determination of each node. For easy understanding, we use figure 4 for example to illustrate the procedure. In figure 4, if node 10 starts the comparison procedure, it first compares the weight with node 5 and 9(the 1-hop neighbors of node 10). If node 10 itself has the largest weight in its 1-hop neighbors at first round of the comparison, it would become a cluster head and the comparison process for node 10 is completed. But here, node 9 has the status “U” and wins the first round weight comparison, so node 9 becomes a candidate node (informed by node 10), and then node 9 compares the weight with node 3, 4, 7 and 8(the 1-hop neighbors of node 9). Again, if node 9 itself has the largest weight in its 1-hop neighbors at the second round of the comparison, it would become a cluster head and the comparison procedure is completed. However, node 3 wins and becomes a cluster head, also finishes the entire comparison procedure of node 10. The rest nodes will become the cluster member of node 3. Actually, in this example, no matter which node begins the clustering procedure, after the comparison procedure, node 3 will always become the cluster head if the status of other nodes were still undetermined.

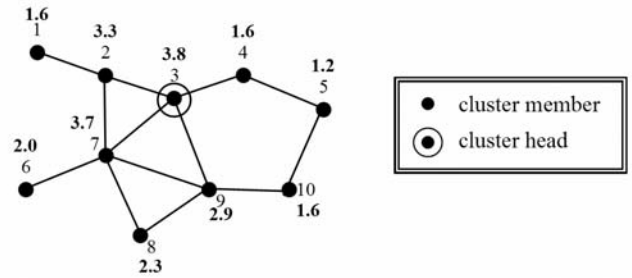

Fig. 4. An example of our clustering algorithm.

Figure 5 is an example of clustering with unidirectional links. The only difference is the weight of each node may be different because of the different weight defined of unidirectional links.

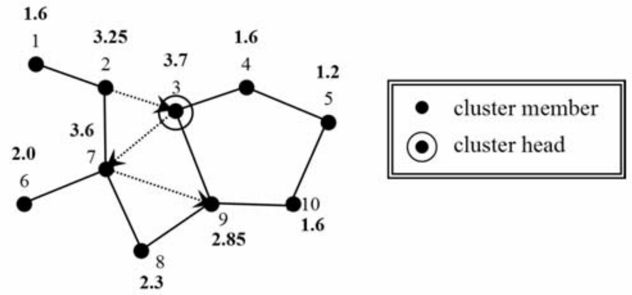

Fig. 5. Clustering with unidirectional links.

Our clustering scheme has several advantages:

1. Pendant nodes are usually not cluster heads.
2. The number of clusters in the network will be reduced.
3. After clustering, the unidirectional links are useful for routing (broadcasting without RTS/CTS).

2.3 Cluster Maintenance Mechanism

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

1. New Link

(1) **New link caused by a new node in the network:** a new node *v* may join the network after clustering procedure finished. Then node *v* would collect the cluster information and choose the node with the smallest *hth()* and smaller than 2 from its 1-hop neighbors, as well as join its cluster. If no such node exists, node *v* itself will become a cluster head and update the cluster information.

(2) **New link between clusters:** when a new link occurs between nodes *u* and *v*, but *dom(u)* and *dom(v)* is different, one of the two nodes may join the other's cluster if the *hth()* could become smaller after joining different cluster, or nodes *u* and *v* would become the gateway nodes of their clusters. Then, update the cluster information.

(3) **New link in the cluster:** when a new link occurs between two nodes in the cluster, the *hth()* and *next()* may change due to the new link, nodes should update the cluster information to make sure the cluster head learns the up to date link state in the cluster.

2. Link Failure

(1) **Link failure between clusters:** in this case, there is no effect upon the topology in clusters. Nodes *u* and *v* only have to check their *gw()*. 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. Figure 6 is an example.

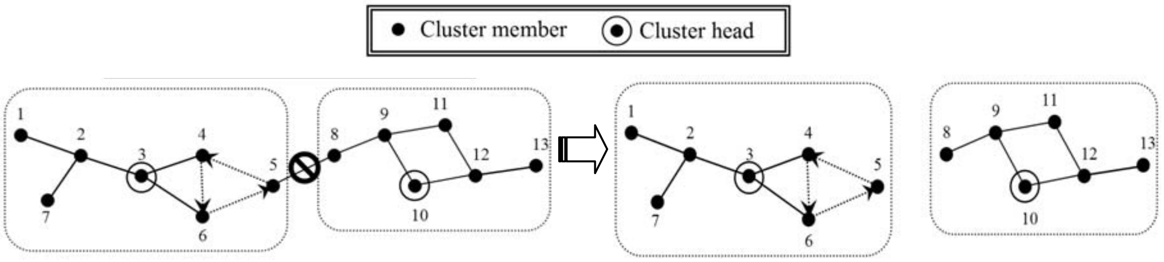

Fig. 6. An example of link change between clusters.

(2) **Link failure in the cluster:** If nodes *u* and *v* belong to the same cluster, then the topology of

the cluster will be changed when the link between *u* and *v* breaks. In the case, nodes *u* and *v* have to notify their cluster head about this change. After receiving the notification, cluster head will update its link states and check if it can still dominate node *u* and *v.* If yes, it does nothing, but nodes *u* and *v* have to update their cluster information. Otherwise, one of the two nodes has to find a new cluster in its' neighbors and join it, or itself becomes a cluster head. Figure 7 is an example.

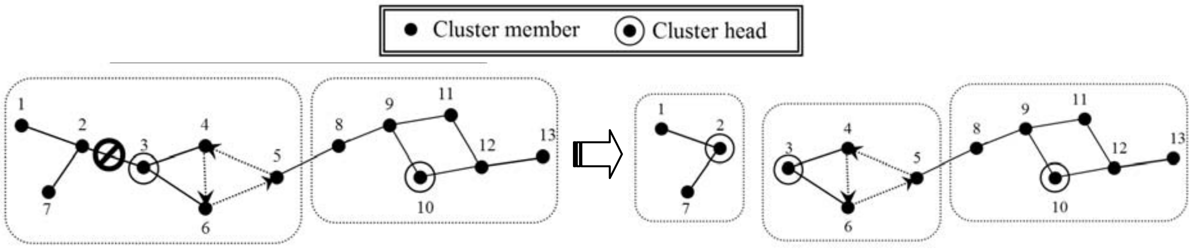

Fig. 7. An example of link change within a cluster.

2.3.1 Cluster head retirement

Most of existing clustering algorithms have the maintenance mechanism. However, the number of clusters will increase constantly after maintaining. Some of them suggest restarting clustering after a period of time, but this reduces the network performance. Here, we provide a retirement mechanism of cluster heads that can enforce some unsuitable cluster heads to become a normal node. Cluster heads will have a chance to become normal nodes, and therefore the network performance can be kept.

If a cluster head *p* detects that:

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

Then, cluster head *p* will retire and join one of the neighboring clusters. Figure 8 is an example of the cluster head retirement.

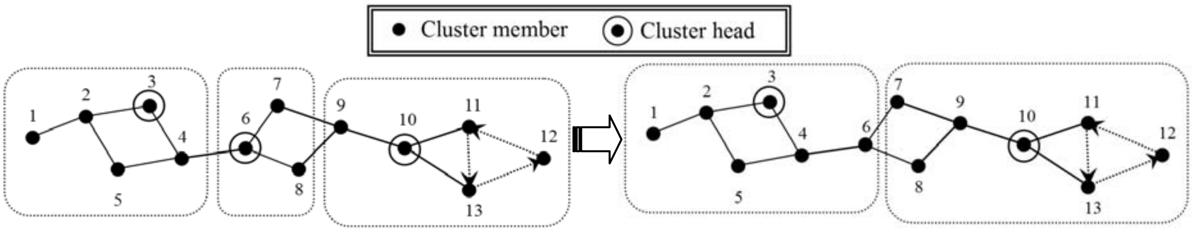

Fig. 8. An example of the retire mechanism.

2.4 Routing Algorithm

In this section, we present a hybrid routing algorithm. This algorithm has two parts: intra-cluster routing and inter-cluster routing. Proactive (table-driven) scheme is used in intra-cluster routing, and reactive (on-demand) scheme is used in inter-cluster routing. Our routing algorithm is an improvement of ZRP [8]. In ZRP, every node in the network has to maintain the *k-hop* information that wastes of overlapping information.

2.4.1 Intra-Cluster Routing

After clustering, cluster heads maintain the link states of the cluster. If a session starts between the source node *u* and the destination node *v*, node *u* first sends a query packet to its cluster head *dom(u)*, and then *dom(u)* will reply node *u* a shortest path between *u* and *v*, thus, node *u* can communicate with node *v* along this path. For example, in figure 9, source node 10 sends a query packet to node 3 first, node 3 then replies node 10 a shortest path 10-9-8-7-6, and therefore node 10 can communicate with node 6 along this path.

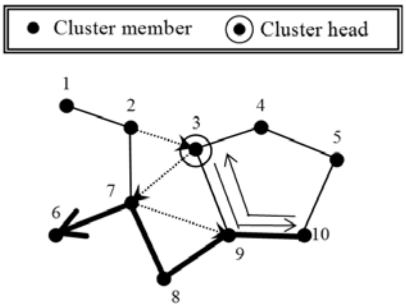

Fig. 9. Routing in a cluster.

2.4.2 Inter-Cluster Routing

When routing among clusters, cluster heads play the role to manage the communications between two clusters. When a cluster head wants to send a message to its neighbor cluster head, first it has to send the message to its gateway nodes, and then the gateway nodes can send the message to other cluster head. In figure 10, node 3 first sends the message to its gateway node 5, node 5 then send the message to its neighbor gateway node 11, thus, node 11 can send the message to its cluster head node 12, and the route can be constructed between two cluster heads.

Now we consider two clusters *u* and *v*. cluster *u* and *v* have to communicate, but cluster *u* has no information about cluster *v,* thus, cluster head *u* will broadcast a route request packet to the gateway nodes in its cluster. To avoid the packet flooding, a sequence number is added to the request packet. Every node only forwards the request packet with the same sequence number once, as DSR. Cluster head *v* will soon receive the route request packet from cluster head *u* and send a reply packet along the route discovery path (use unidirectional neighbor table and link layer broadcast to pass the incoming unidirectional links). When a cluster head on this path receives the route reply packet, it will shorten this path by its own link state information and then continues the route reply. Thus, node *u* finally obtains a route to *v.*

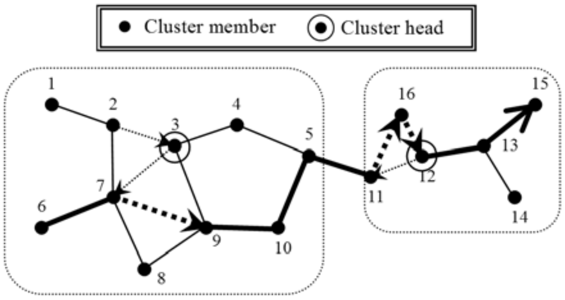

Fig. 10. Routing between clusters.

2.4.3 Route Maintenance

The routing path may become disconnect due to the node mobility. Thus, we need an efficient route maintenance mechanism to ensure the success of routing. For example, in figure 11, the route 1-2-3-6-5-12-13-15-16-17 is established from node 1 to node 17. If the link between nodes 5 and 12 breaks, node 5 will notify its cluster head 3 this fact, node 3 will send a message to its gateway to find another path to connect the cluster 15, and obtain the repaired path 1-2-3-4-7-8-11-12-13-14-16-17.

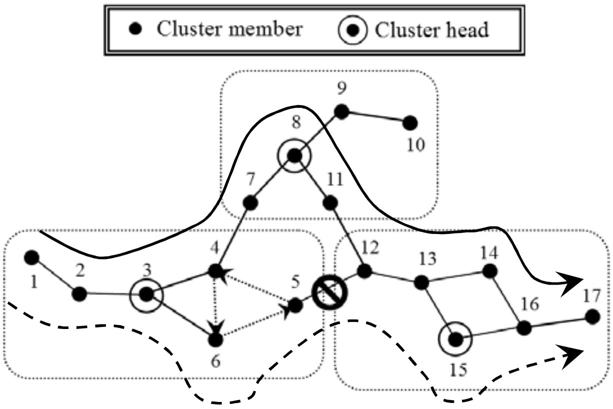

Fig. 11. The route maintenance mechanism.

III. Simulation Results

In this section, we compare our clustering scheme with 2-hop Least Cluster Change (LCC) using Lowest-ID [6], 2-hop SK-clustering [21] by some metrics such as the number of clusters and the stability. We also compare our routing scheme with DSR [2] and ZRP [8]. The simulation areas including $1000m \times 1000m$, $2000m \times 2000m$ and $2200m \times 600m$ (for routing). The number of nodes is from 50 to 500 and the transmission ranges is vary from 100*m* to 300*m*, simulation time is 300 seconds, speed of nodes is 0 to 30*m/s* with random way point mobility model. We change the following two special parameters to create the unidirectional network environment. First, we define two different transmission ranges for nodes. It means that there are some nodes in the network have smaller transmission range. Second, we control how many nodes to have a smaller transmission range (the percentage of unidirectional nodes).

3.1 Analysis of the cluster scheme

In the simulation of clustering algorithm, we consider the following parameters: (1) *Number of cluster* – in the cluster-based routing protocol, the overhead mainly comes from the clustering procedure and the communication between cluster heads. Therefore, the fewer number of cluster heads may cause the lower overhead; (2) *Number of role change* – a node's status changes from a cluster head to a member node or from a member node to a cluster head is the role change. This is a measurement of cluster stability. (3) *Number of cluster switch* – a cluster switch means a member node changes its cluster head from one to another. This is also to measure the stability of a cluster algorithm. A stable algorithm can save the resources of the network. Thus we believe that a good clustering algorithm should avoid the frequently role changes or cluster switches with changing topology. Figure 12 and 13 show the performance of the number of cluster in the environment with unidirectional links. Our scheme provides superior performance in the number of clusters when the percentage of unidirectioanl links grows. Figure 14 and 15 show that even in the bidirectional network environment, our scheme still outperforms the others in the number of clusters, only the LCC pays a lot of maintenance overhead to keep the number of cluster low (figure 16 to 19).

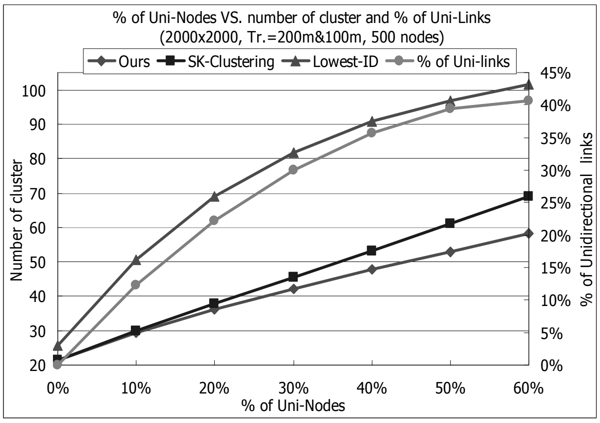

Fig. 12. % of unidirectional nodes vs. number of cluster and % of unidirectional links.

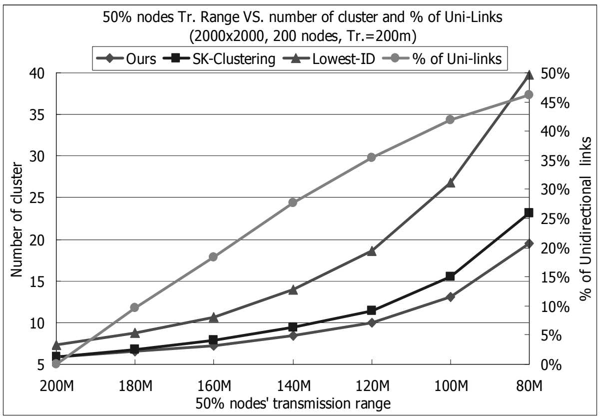

Fig. 13. 50% nodes Tr. range vs. number of cluster and % of unidirectional links.

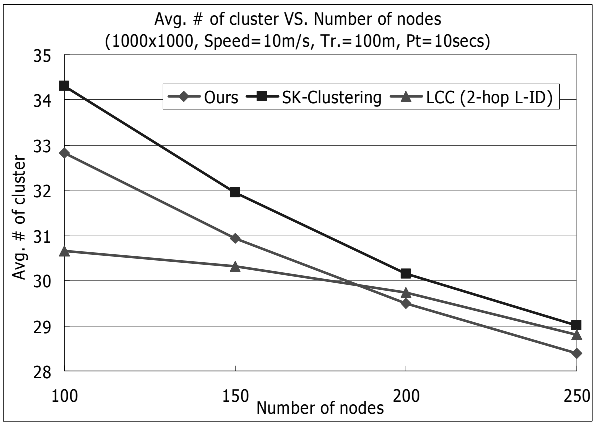

Fig. 14. Average number of cluster vs. number of nodes.

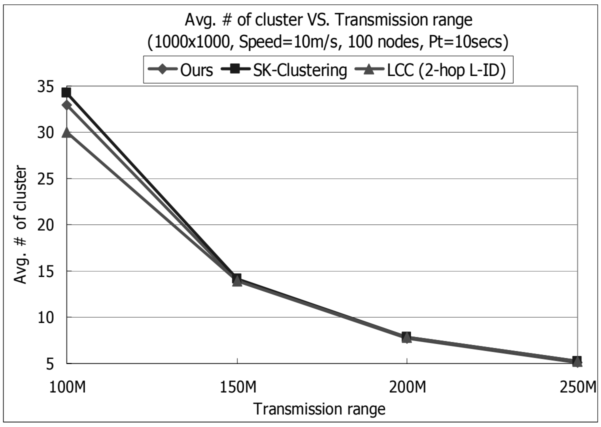

Fig. 15. Average number of cluster vs. transmission range.

The metrics of role changes and cluster switches are shown from figure 16 to 19. Simulation results show that our scheme performs well in both role changes and in cluster switches. As expected, in figure 17 and 19, higher mobility causes higher role changes and cluster switches.

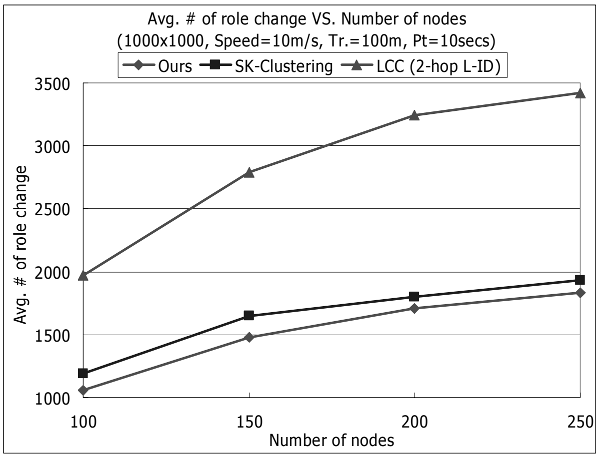

Fig. 16. Average number of role change vs. number of nodes.

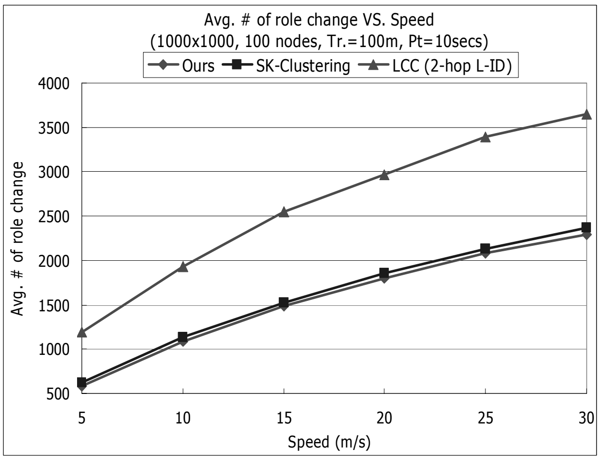

Fig. 17. Average number of role change vs. speed.

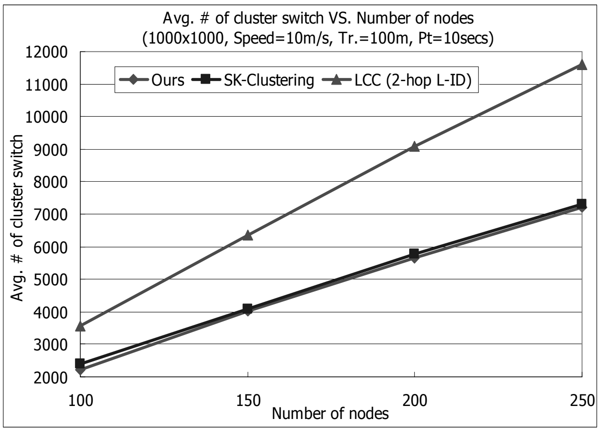

Fig. 18. Average number of cluster switch vs. number of nodes.

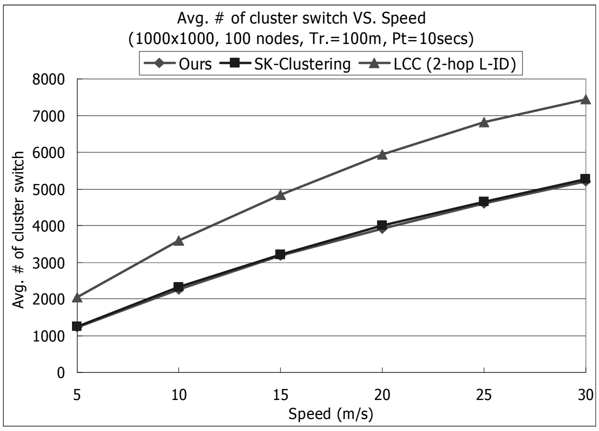

Fig. 19. Average number of cluster switch vs. speed.

3.3 Analysis of routing protocol

In the course of this experiment, the following parameters are measured: (1) *Delivery ratio* – ratio of the data packets delivered to the destination to those generated by the source (1 packet per second). (2) *Number of RREQ forwarding nodes* – when a source broadcasts the RREQ packet, the number of nodes that help re-broadcast the RREQ packet. (3) *Average hop count* – the average hop count form the source to the destination. Figure 20 and 21 show the delivery ratio of our routing algorithm compares to the ZRP and DSR. Because of our route maintenance is based on the cluster information update, the delivery ratio of our cluster routing protocol (CRP) drops when the mobility is high. On-demand routing scheme such as DSR, source node has to broadcast a route request packet to the entire network when discovering a route. Blind flooding of RREQ packets cause the waste of bandwidth. In figure 22 and 23, the cluster scheme do not use flooding thus results lower forwarding nodes. Although ZRP uses border-casting to prevent flooding. When the network connectivity is high, the border-casting still floods the RREQ to almost entire network. The average hop count is shown in figure 24 and 25. As expected, three routing protocols finds short paths but DSR is the shortest one.

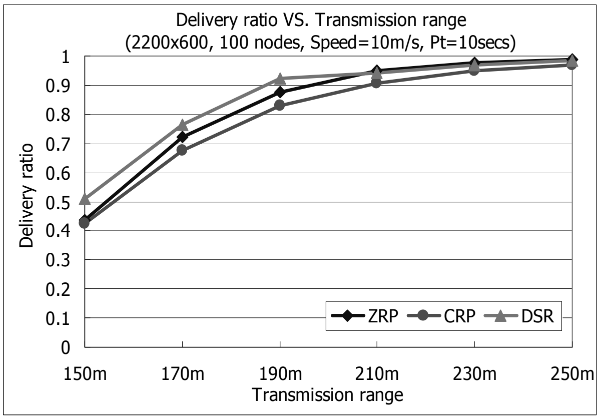

Fig. 20. Delivery ratio vs. transmission range.

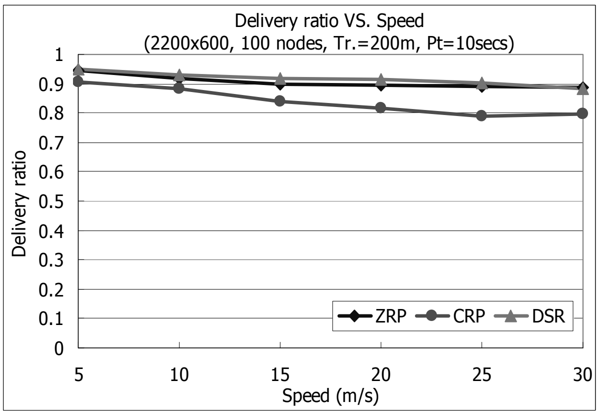

Fig. 21. Delivery ratio vs. speed.

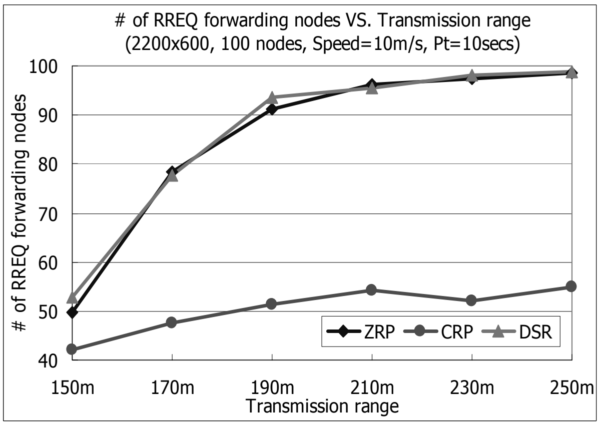

Fig. 22. Number of RREQ forwarding nodes vs. transmission range.

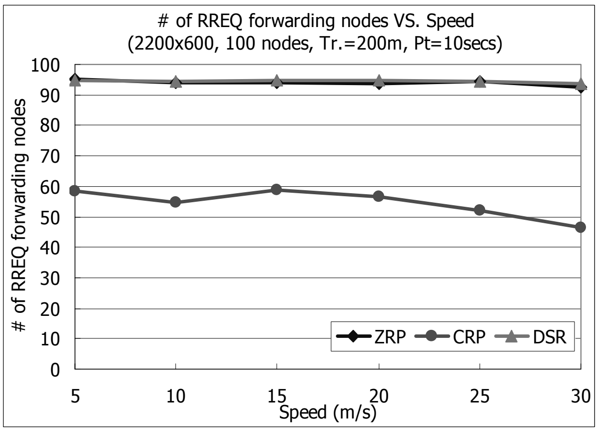

Fig. 23. Number of RREQ forwarding nodes vs. speed.

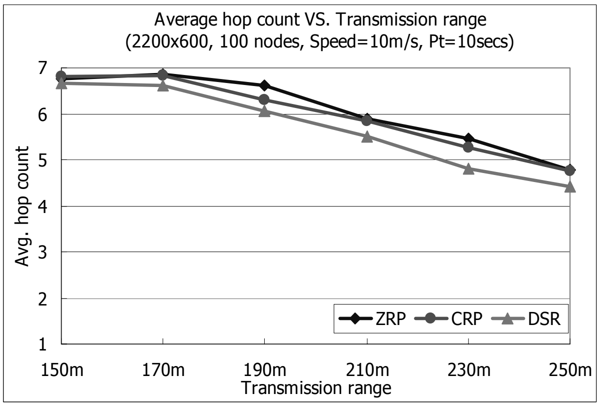

Fig. 24. Average hop count vs. transmission range.

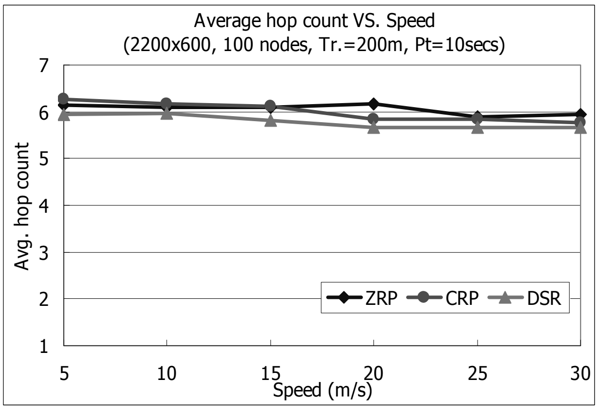

Fig. 25. Average hop count vs. speed.

IV. Conclusions

Ad hoc networks are an increasingly promising area of research with practical applications. It is very suitable for emergency, temporary and wireless network environment. In the future network environment, the different devices may use different transmission range for communications thus results in the unidirectional links in the network. We propose an efficient clustering routing scheme that supports unidirectional network environment. Simulation results show that our scheme has better performances. Less numbers of role changes and cluster switches indicate the network is more stable in our scheme. Additionally, the performance of routing is well. Especially, our scheme uses less resource to find a route than DSR and ZRP. We plan on improving our maintenance schemes and some optimizations as our future works.

Reference

[1] S. H. Bae, S.-J. Lee, W. Su, and M. Gerla, “The design, implementation, and performance evaluation of the on-demand multicast routing protocol in multihop wireless networks,” *IEEE Network*, vol. 14, no. 1, pp. 70-77, 2000.

[2] J. Broch, D. B. Johson, D. A. Maltz, “The dynamic source routing protocol for mobile ad hoc networks,” *Mobile Ad-hoc Networks (manet)*, http://www.ietf.org/html.charters/ manet-charter.html, Aug. 1998.

[3] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” *Proc. the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1998, pp.85-97.

[4] C. C. Chiang and M. Gerla, “Routing and multicast in multihop, mobile wireless network,” *Proc. ICUPC '97*, 1997.

[5] C. C. Chiang, H. K. Wu, W. Liu and M. Gerla, “Routing in cluster multihop, mobile wireless networks with fading channel,” *Proc. IEEE Singapore International Conference on Network (SICON’97)*, 1997.

[6] M. Gerla, G. P. and S.-J. Lee. “Wireless, mobile ad-hoc network routing,” *IEEE/ACM FOUCUS’99*, 1999.

[7] M. Gerla and J.T.-C. Tsai, “Multicluster, mobile, multimedia radio network,” *ACM Journal on Wireless Networks*, pp. 255-265, 1995.

[8] Z. J. Haas and M. R. Pearlman, “The zone routing protocol (ZRP) for ad hoc networks,” *Mobile Ad-hoc Networks (manet)*, http://www.ietf.org/html.charters/ manet- charter.html, Nov. 1997.

[9] T.-C. Hou, T.-J. Tsai, “Adaptive clustering in a hierarchical ad hoc network,” *Proc. ICS Workshop on Computer Networks, Internet, and Multimedia*, 1998.

[10] C.-S Hsu, Y.-C Tseng, and J.-P Sheu, ”An efficient reliable broadcasting protocol for wireless mobile ad hoc networks,” *Proc. 7th Mobile Computing Workshop*, pp. 189-201, 2001.

[11] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, “Scalable routing strategies for ad hoc wireless networks,” *IEEE Journal on Selected Area in Communication*s, vol. 17, no. 8, Aug. 1999.

[12] M. Joa-Ng and I.-T. Lu, “A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks,” *IEEE Journal on Selected Area in Communications*, vol. 17, no. 8, Aug. 1999.

[13] D. Kim, S. Ha, Y. Choi, “Cluster-based dynamic source routing protocol in wireless ad-hoc packet radio network with variable-sized clusters and variable transmission ranges,” *Proc. Vehicular Technology Conference*, vol. 3, pp. 2441-2445, 1999.

[14] D. Kim, S. Ha, and Y. Choi, “K-hop cluster-based dynamic source routing in wireless ad-hoc packet radio network,” *Proc. Vehicular Technology Conference*, vol.1, pp. 224-228, 1998.

[15]T-.J. Kwon, M. Gerla, “Clustering with power control,” *Proc. Military Communications Conference*, vol. 2, pp. 1424-1428, 1999.

[16] H.-C. Lin , Y.-H. Chu, “A clustering technique for large multihop mobile wireless networks,” *Proc. Vehicular Technology Conference*, vol. 2, pp. 1545-1549, 2000.

[17] C.-B. Lin and S.-F. Hwang, “An efficient routing algorithm in ad hoc networks,” in *Proc. National Computer Symposium*, pp. C494-C503, 1999.

[18] D. A. Maltz, J. Rroch, 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,no. 8 Aug 1999.

[19] E. M. Royer and C.-K. Toh, “ A review of current routing protocols for ad hoc mobile wireless networks,” *IEEE Personal Communications*, vol. 6, pp. 46-55 2, April 1999.

[20] P. Sinha, R. Sivakumar and V. Bharghavan, “CEDAR: A core-extraction distributed ad hoc routing algorithm,” *Proc. IEEE INFOCOM’99*, pp. 202-209.

[21] 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,” *Proc. 7th Mobile Computing Workshop*, pp. 175-181, 2001.