An Effective Approach to Improving Packet Delivery Fraction of Ad Hoc Network

Jiahong Wang, Takaaki Mikami, Kazuki Kanamori, Eiichiro Kodama, and Toyoo Takada

***Abstract*—This paper addresses the subject of improving performance of an ad hoc network system suffering from high resource contention by improving its packet delivery fraction. The suggested system model assumes that nodes can be divided into two classes: the servers that provide services and the clients that request services provided by servers. It is found that for such a system, it is the server and its adjacent nodes that dominate performance, and tend to cause bottlenecks due to congestion. To reduce the possibility of bottleneck occurrences, it is suggested that resource contention of the area where a server node is located be relieved. This is achieved mainly by decoupling route-requesting from service-providing activities, and reducing the former. On the one hand, for each busy, and thus critical server, a node cluster is created by constructing a server-centered connected dominating set to actively reduce contention level. On the other hand, routes are adaptively selected to avoid high congestion area. It is found that in this way, the overall data packet delivery fraction can be improved greatly. Results of simulation experiments demonstrated the effectiveness of the proposed approach.**

***Index Terms*—ad hoc network, server-centered connected dominating set, resource contention, packet delivery fraction.**

I. Introduction

An ad hoc network is an autonomous collection of mobile nodes that communicate over wireless links. Each node functions as both a terminal and a router, i.e., the routing functionality is its constituent element. Each node is free to move in any direction at any time, so the network's topology may change unpredictably over time.

For many applications on ad hoc network, nodes can be divided into two classes according to their roles playing in applications: the server that provides services and the client that requests services provided by servers. For this, an example is given below.

Example 1 (client-server environment on ad hoc network)**.** *Assume a crowded ad hoc network with a lot of nodes, where a teacher and a class of students, each one being with a mobile computer, are taking an outdoor seminar. The teacher's computer is used to coordinate the seminar, and to take the responsibility for storing data used, questions raised, and reports submitted by students. Obviously that computer would play the role of a server, and others would be clients.*

It was found that for such a system, it is the server and its adjacent nodes that dominate performance, and tend to cause bottlenecks due to congestion. To reduce the possibility of bottleneck occurrences, we suggest relieving resource contention of the area where a server node is located. In fact, no matter it plays a role of server or client, each node has to assist other nodes with discovering routes. Then in a service intensive environment such as that given in Example 1, this extra task enforcing on a server node would degrade system performance [1]. Especially, if reactive (on-demand) routing protocols such as AODV (Ad hoc On-Demand Distance Vector) [2] is used, the performance down would become more serious.

The authors are with the Faculty of Software and Information Science, Iwate Prefectural University, Takizawa, Iwate, 020-0193 JAPAN, e-mail: wjh@iwate-pu.ac.jp.

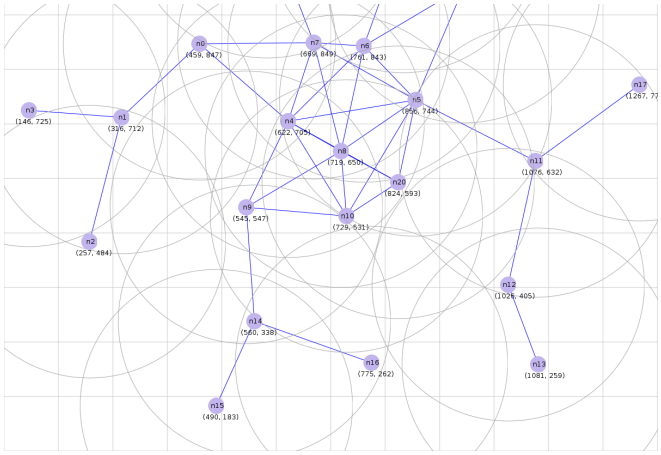

Fig. 1. An example for showing how the bottleneck can occur in an ad hoc network. In this example, the Node 8 is functioning as a server, and others are clients.

To further explain the problem, let us consider the network shown in Fig. 1. Assume that node 2 is to exchange data with node 17. If there is not any pre-hand knowledge about the route from node 2 to node 17, all nodes around node 8, the server node, has to actively take part in the process of discovering routes. This may burden on server node too much. This also causes another serious problem: more collision coming from interference. It was said that due to interference, end-to-end throughput would be as low as only 4.2% the link data rate in the worst case [3]. As results, performance of the server, and therefore the system, would be affected seriously.

Then in order that a system can give a satisfactory performance even in the case of high resource contention, we have to know which nodes are playing the role of servers. For this, we will propose an approach to "mining" the server nodes on demand. When server nodes are located, we have to ease the resource contention around them. For this, topology control based on transmission power reduction [4]–[6] and dominating-set-based routing [7]–[12] seem to be promising solutions. We restrict ourselves to the latter.

Dominating-set-based routing is such a solution that selects certain nodes from the network as gateway nodes. These gateway nodes form a Connected Dominating Set (CDS) and are responsible for routing within network. We suggest constructing CDSes centered around server nodes, so that other nodes of a CDS can take the place of the server node

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

to conduct routing, to prevent a server node from excessive routing activities.

This paper addresses the subject of improving performance of an ad hoc network system suffering from high resource contention by improving its packet delivery fraction when necessary. It is suggested that resource contention of the area where a server node is located be relieved. This is achieved mainly by decoupling route-requesting from service-providing activities, and reducing the former around the servers. On the one hand, for each busy, and thus critical server, a node cluster is created by constructing a server-centered CDS to actively reduce the level of resource contention. On the other hand, routes are adaptively selected to avoid high congestion area. It is found that in this way, the overall packet delivery fraction can be improved greatly. Results of simulation experiments demonstrate the effectiveness of the proposed approach.

The remainder of this paper is structured as follows. In Section II, we describe the model of computation and formally define the problem. An algorithm for solving the problem defined in Section II is given in Section III. Results of a performance study is discussed in Section IV. Finally, Section V concludes the paper and indicates directions for future research.

II. Problem Statement

The problem to be solved in this research, as described in Section I, is formulated as follows.

Problem Statement. *Given an ad hoc network of mobile nodes, in which each node can be categorized as either server or client. An approach is to be devised to improve the overall system performance by relieving working load of the server nodes when necessary. Taking Fig. 2 as an example, this means that when number of source nodes reaches about 30, some means are taken to avoid the sudden falling of packet delivery fraction shown in this figure.*

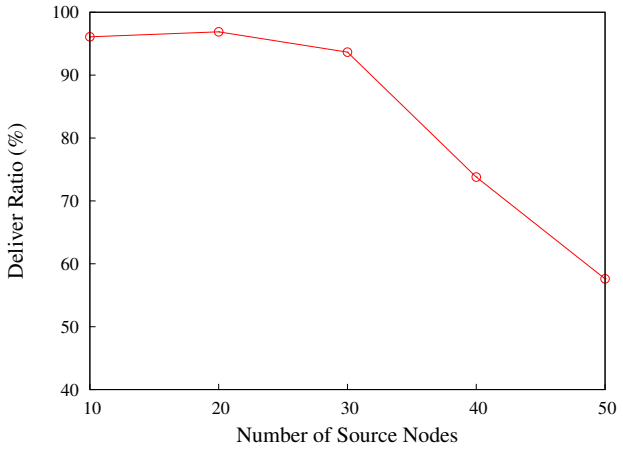

Fig. 2. Simulation results showing that packet delivery fraction may fall suddenly due to resource contention.

A possible solution to this problem is to construct a "global" CDS that covers the whole ad hoc network. This dominating-set-based approach is widely used for constructing network backbones. However, it could be very expensive to maintain such a CDS dynamically to adapt topology changes due to the mobility of mobile nodes, which is a main property of ad hoc networks. It is true that various low cost approaches have been proposed [11], [13]–[20]. For the defined problem, however, these approaches could not be very helpful. It is a lightweight and diameter-limited CDS that is required here. On the one hand, as stated above, existing dominating-set-based approach requires exchanging a lot of messages to construct and maintain a CDS, which contradicts our objective: ease the resource contention. On the other hand, considering our objective, to limit the diameter of a CDS to some specified threshold, as proposed in [21]–[24], is necessary. "With the help of a CDS with small size and diameter, routing is easier and can adapt quickly to topology changes of a network" [21]. Differently, we require that a CDS should be centered at a server node, and its structure should be suitable for the defined problem. That is, we focus our attention on the construction of a service-providing CDS. More attention should be paid to the internal structure of the CDS.

III. Relieving Load of the Critical Nodes

In this section, we first provide notations, definitions, and some fundamental concepts used in the subsequent sections. Then we propose an approach to identifying server nodes and relieving their loads. Figure 3 shows the system model, and Figure 4 shows the resulting CDS corresponding to Fig. 3. We will use these two figures to give our description.

A. Definitions and Preliminaries

Let graph $G = (V, E)$ represent an ad hoc network, where V consists of all nodes and E represents all the links. Without losing generality, the nodes in V are assumed to be located in a plane, and each node $v_i \in V$ has a transmission range r. An edge $(v_i, v_j) \in E$, iff $d(v_i, v_j) < r$, where $d(v_i, v_j)$ denotes the distance between v_i and v_j.

Firstly we define the term *server node*. For any node $v \in V$, the number of inbound route requests to v for time period $T = [t_0, t_1]$ is denoted as *IN(v, T)*, and the number of outbound route requests originating from v is denoted as *OUT(v, T)*. The number of inbound route requests, including forwarded ones, made by nodes within 3 hops from v (i.e., locally) is denoted as $IN_{\leq 3}(v, T)$, and that made by nodes 4 hops away (i.e., globally) is denoted as $IN_{>3}(v, T)$. The number of packets received and sent successfully is denoted as *TX_Rcv(v, T)*. The number of lost packets due to collision and congestion is denoted as *Coll_Cong(v, T)*. Then server node is determined according to the following definition.

Definition 1 (Server node)**.** *A node $v \in V$ is taken as a server node for time period of T iff it satisfies the following three conditions, where λ_1, λ_2, and λ_3 are threshold values defined by users beforehand.*

1) *v satisfies the following inequality, and thus is considered as being functioning as a service provider,*

$$\frac{IN(v,T)}{IN(v,T)+OUT(v,T)} \geq \lambda_1$$

2) *v satisfies the following inequality, and thus is considered as being involved in processing route requests*

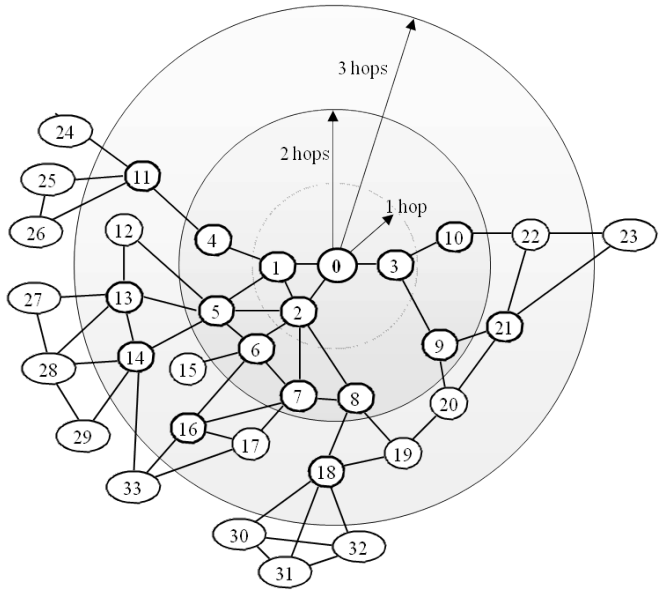

Fig. 3. System structure. Node 0 is a server node, and others are clients.

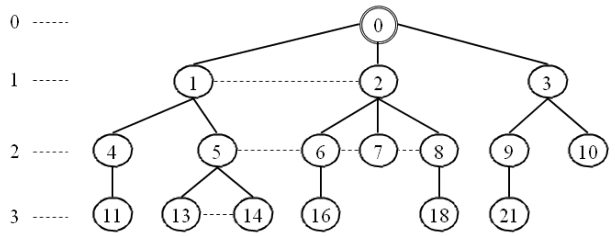

Fig. 4. Server-centered CDS would be a tree that takes a server node as its root. Numbers in circles are node ids. Numbers to the left are level numbers.

coming from global nodes.

$$\frac{IN_{>3}(v,T)}{IN_{\leq 3}(v,T)+IN_{>3}(v,T)} \geq \lambda_2$$

3) *v satisfies the following inequality, and thus is considered being suffering from resource contention.*

$$PPLost = \frac{Coll\_Cong(v,T)}{TX\_Rcv(v,T)+Coll\_Cong(v,T)} \geq \lambda_3$$

Figure 3 shows a part of an ad hoc network. Node 0 is a server node. Nodes 23-33 are taken as *global nodes* and others are taken as *local nodes*.

Next we define the term *server-centered CDS*. A set of nodes $V' \subseteq V$ forms a Connected-Dominating Set (CDS) of network G, if all members in $V - V'$ are neighbors of a node in V' and the subgraph $G[V']$ induced from V is connected. A server-centered CDS, denoted as *SCDS*, is a CDS that satisfies the following definition.

Definition 2 (Server-centered CDS: SCDS(v))**.** *Given a server node v, the v-centered CDS, denoted as SCDS(v), is a CDS of diameter 3. It takes v as its kernel for providing services, layer 3 as its interface for accepting requests, and layers 1 and 2 as the buffering layer for resisting interferences from interface layer and outer nodes, and forwarding packets.*

In Fig. 3, nodes represented by bold black circles form a server-centered CDS, the *SCDS(0)*.

TABLE I
NOTATIONS USED IN THIS PAPER

notations	meanings
$x.id$	identity number of node x
$x.level$	level number of node x
$x.father$	identity number of node x's father
$v.adj$	adjacent nodes of node x
$x.sons$	sons of node x
$path\_to(s)$	route to server node s
$Get\_F(v)$	return id of v's father node
$sender(p)$	return id of the node that sends packet p
$F(v)$	fathers of node v
$broadcast(m,f,p)$	broadcast message m, id f of the current node's father, and path p to the server node

The remainder of this section gives a property of the server-centered CDS. We adopt the definition of d-CDS in [25] as follows. G_d denotes the $d\text{-}closure$ of G. It has the same node set as G, but the edge set is: $\forall u, v \in V$, there is an edge connecting u and v iff $0 < \delta(u,v) \leq d$, where $\delta(u,v)$ is the distance between u and v. A set $V' \subseteq V$ is a *d-hop* dominating set of G if it is a dominating set for G_d. That is, if every node of G is within a distance d of some node in V'. A *1-dominating set* is simply called a dominating set. We say that V' is *d-hop connected* if it is connected in G_d.

Then according to the above definition of d-CDS, and referring to Fig. 2 and Fig. 3, it is obvious that the following property holds true for a server-centered CDS. More description will be given in Section III-B.

Property 1. *A server-centered CDS is a 1-CDS.*

B. The Algorithm SCDSR: SCDS-based Routing

This section gives an approach to solving the above defined problem, including functions for collecting system status data (Func. 1), constructing server-centered CDS (Func. 2), forwarding packets (Func. 3), and avoiding busy routes (Func. 4). Table 1 summarizes the notations that will be used.

1) Determining the server node: For every node in an ad hoc network, data concerning system status defined in Definition 1 will be collected periodically according to the algorithm shown in Func. 1 below. These data will be used to decide which node is a server node, and when to start constructing server-centered CDS. The parameters λ_1 - λ_3 in Definition 1 can be determined beforehand by considering characteristics of the application environment. For example, one can set λ_1, λ_2, and λ_3 to 0.9, 0.5, 0.3, respectively.

2) Constructing server-centered CDS: When a node becomes a server node according to Definition 1, it is the time to construct the server-centered CDS. An algorithm is given in Func. 2.

When a server-centered CDS is constructed, each node of it will know the path to the server node s. For example, in Fig. 4, node 13 knows the route to node 5, 1, and 0.

Note that two nodes apart from no more than 3 hops may start constructing server-centered CDS at the same time. At that time, we may simply stop constructing process of the

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Function 1 Get system status

Require: Time interval T
Ensure: Each node knows system status within 3 hops
1: **for** (each node $v \in V$) **loop**
2: **for** (each time interval T) **do**
for (each sent, received, and lost packet) **update**
· *IN*, *OUT*,
· $IN_{\leq 3}$, $IN_{>3}$,
· *TX_Rcv, Coll_Cong*
3: **end for**
4: **end for**
5: **end for**

node with a smaller node id to resolve the confliction. We may also compare the metrics defined in Definition 1 to determine to stop which one more smartly.

A server-centered CDS constructed with using Func. 2 will be a 1-CDS. This is because the interior nodes of the tree constructed in Func. 2 are taken as the CDS nodes, and all leaf nodes are removed.

3) SCDS-based routing: Without losing generality, we assume that the method of discovering a route is the same as

Function 2 Construct SCDS

Require: A server node s identified by s_{id}
Ensure: A s-centered CDS, *SCDS(s)*
1: **for** (each $v \in V$) **do** $v.level \longleftarrow \infty$;
2: $s.level \leftarrow 0$;
3: *SCDS(s)* $\leftarrow$ { $<s.id, s.level>$ };
4: $m_0 \leftarrow <s.id, s.level>$;
5: *broadcast*(m_0, *null*, *null*);
6: **for** (each $v \in V$ receiving m_0 and $v.level = \infty$) **do**
7: $v.father \leftarrow s.id$;
8: $v.level \leftarrow 1$;
9: $m_1 \leftarrow <v.id, v.level>$;
10: *broadcast*(m_1, $v.father$, *path_to(s)*);
11: **end for**
12: $l \leftarrow 1$;
13: **repeat**
14: **for** (each $v \in V$ receiving m_l) **do**
15: **switch** ($v.level$) {
16: **case** $l-1$:
17: **if** ($get\_F(sender(m_l)) = v$)
18: **then** $v.sons \leftarrow v.sons \cup \{ sender(m_l) \}$;
19: **case** l:
20: **if** ($l > 3$) **then** break;
21: $v.adj \leftarrow v.adj \cup \{ sender(m_l) \}$;
22: **case** ∞:
23: **if** ($l > 3$) **then** break;
24: $v.level \leftarrow l+1$;
25: $m_{l+1} \leftarrow <v.id, v.level>$;
26: *father* $\leftarrow min\_id(F(v))$;
27: *broadcast*(m_{l+1}, *father*, *path_to(s)*);
28: **endswitch**
29: **end for**
30: $l \leftarrow l+1$;
31: **until** ($l > 4$);
32: *SCDS(s)* $\leftarrow$ *SCDS(s)* $\cup$ { v }$_{v.sons \neq null}$;

Function 3 SCDS-based routing

Require: *SCDS(s)* and an inbound route request packet p
Ensure: The request is processed within *SCDS(s)*
1: **if** (p is for server node s) **then**
2: send p to s along a *SCDS(s)* path to s directly
3: **else**
4: **repeat**
5: **try** (forward p to) {
6: · a level-3 node;
7: · a level-2 node if no adjacent level-3 nodes;
8: }
9: **if** (no level-2 and -3 nodes are available)
10: **then** break;
11: **else if** (find a route, say r) return(r);
12: **until** (p has traveled k level-1 subtrees);
13: broadcast p and find route within the remaining level-1 subtrees using the general method as AODV;
14: **end if**

that of AODV, except a route request is in the server-centered CDS, in which case it is processed according to Func. 3.

The idea of Func. 3 is that route requests that are not for nodes within the CDS are forwarded along the outer layer as far as possible so as to avoid interference to the server node. However, if there is no such nodes available, a shortest path across the server node is taken so that the packet can get out of the CDS as soon as possible.

The optimal value of parameter k in Func. 3 is determined to be 2 by experiments. We had expected that a larger k would give better performance, but by experiments we found that a k larger than 2 would cost too much.

4) Avoiding the busy routes: While a node is exchanging data with some other node using a known route, a new route may become available to it. In that case, we have to make decision which one should be used to get better performance such as higher throughput, higher delivery fraction, and so forth. This decision is made according to Func. 4. By experiments it was found that the optimal value of parameter α is between 0.7 to 0.8, and was set to 0.75 in simulation experiments.

Function 4 Process a route response

Require: A packet responding to a route request
Ensure: A more suitable route is elected.
1: When a packet responding to a route request goes back to the node that has made the request, let the route be $<v_0, v_1, \cdots, v_n>$, it collects data about $PPLost_i$, and computes the survival probability of a data packet that will be sent along the route as follows:

$$PPSurv = \prod_{i=1}^{n} (1 - PPLost_i)$$

2: **if** (*PPSurv* of the current route $\leq \alpha$ and *PPSurv* of the new route $>$ that of the current route) **then** replace the route with the new one;

IV. Performance Evaluation

Simulation experiments have been done to study performance of the proposed approach. We take the AODV

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

algorithm, which is known to be of very high packet delivery fraction, as a baseline, and evaluate to what extent the proposed approach can improve system performance in the condition of high resource contention. This section describes simulation environment, presents and discusses simulation results.

A. Simulation Environment

Simulation experiments are done using Network Simulator-2. As shown in Fig. 5, the field configuration is 2000 m $\times$ 1500 m square with total 100 nodes. Nodes are randomly placed on the plane. Continuous Bit Rate (CBR) traffic sources are used for traffic model. Data packet size is 512-byte. The number of source-destination pairs is varied to change the offered load. The number of source nodes is 10, 20, 30, 40, and 50 separately. The MAC we employ is 802.11 MAC.

Three performance metrics, as described below, are used for the performance study:

- Packet Delivery Fraction: The ratio of total data packets successfully received to total ones sent by CBR sources.
- Average Path Length: The average length of all paths that connect source and destination nodes.
- Number of Router Drops: Total number of data packets dropped by the routers. This routing load metric demonstrates the level of resource contention and also evaluates the efficiency of the routing protocol.

In order to more clearly demonstrate the effectiveness of the proposed approach, we disable the mobility of nodes in this performance study. For the same reason, traffic model is so designated that the node 0 in Fig. 5, located at the center-left of the figure, could be always selected as the server node.

B. Experimental Results and Discussions

Figure 6 gives results of packet delivery fraction vs. number of source nodes. Line AODV-30 (resp. AODV-50) is for the original AODV in the case that 30% (resp. 50%) accesses from source nodes are for the server node. Lines SCDSR-30 and SCDSR-50 are for the proposed approach.

From Fig. 6 we know that the proposed approach can effectively improve packet delivery fraction in the case of high resource contention. In the case of 30% (resp. 50%) accesses going to a server node, the proposed approach

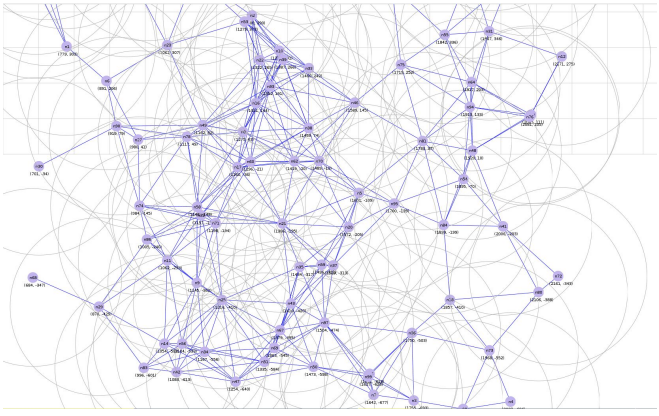

Fig. 5. The ad hoc network used in performance study.

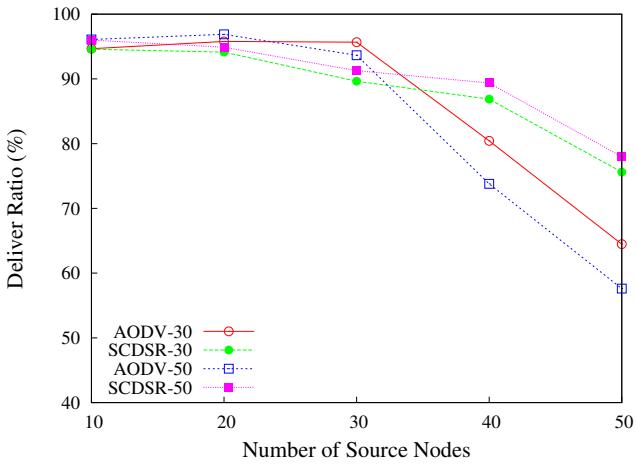

Fig. 6. Results of Packer delivery fraction vs. number of source nodes

begins to outperform the original AODV when number of source nodes becomes larger than 35 (resp. 31), and raises packet delivery fraction from about 64% (resp. 60%) to about 75% (resp. 80%) when number of source nodes is 50. From the same figure we also know that if the level of resource contention is low, the original AODV would give higher performance with respect to the packet delivery fraction.

We think that the main reason of the results shown in Fig. 6 lies in the proposed approach's ability to reduce resource contention, which is demonstrated in Fig. 7 and 8. Figure 7 shows that, numbers of packets dropped at router nodes are lower for the proposed approach, implying that the proposed approach really has the ability to reduce resource contention. As shown in Fig. 8, when the level of resource contention becomes high, the average lengths of paths become long due to the adaptive route redirection mechanism of AODV to avoid the congestion-intensive routes. This increasing average path in turn increases the packet delivery fraction. Compared with the original AODV, the proposed approach tends to reduces the resource contention, and thus the average length of paths.

V. Conclusions

We have identified a performance problem in an ad hoc client / server system, and given a solution of it. The suggested system model divides nodes into two classes: servers and clients. For each server, a server-centered CDS (SCDS) is constructed. Each SCDS has a kernel, an interface layer, and a buffering layer. The CDS with such a structure, combined with the supporting algorithms, is found to be able to relieve load of the server node and avoid interference from outer nodes, and thus improve overall system performance. Simulation experiments have been conducted and experiment results demonstrated effectiveness of the proposed approach.

An unsolved problem is that a SCDS cannot accurately represent the physical distance. So interference may still reach server node. Solving this problem is our future work.

References

[1] Y. Yi, M. Gerla, and T. J. Kwon, "Efficient flooding in ad hoc networks: a comparative performance study," in *Proc. IEEE International Conference on Communications (ICC'03)*, Anchorage, USA, May 2003, pp. 1059–1063.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

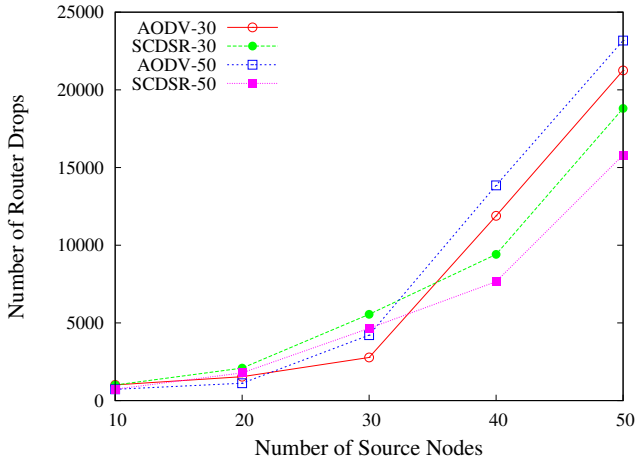

Fig. 7. Results of number of router drops vs. number of source nodes

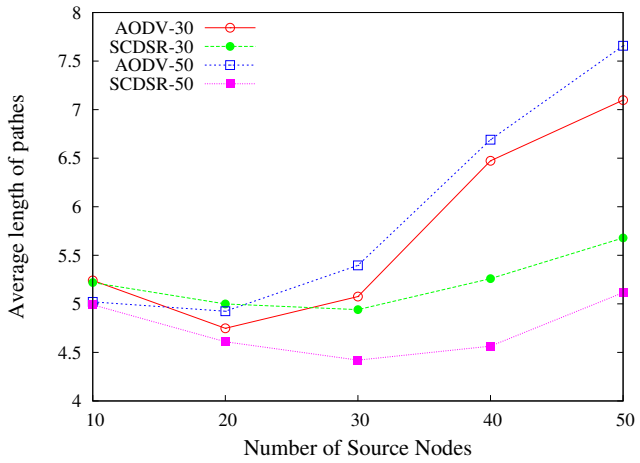

Fig. 8. Results of average length of paths vs. number of source nodes

[2] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proc. IEEE 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, 1999, pp. 90–100.

[3] R. Mangharam and R. Rajkumar, "Max: A maximal transmission concurrency mac for wireless networks with regular structure," in *Proc. IEEE International Conference on Broadband Communications, Networks and Systems (BROADNETS'06)*, San Jose, CA, 2006, pp. 1–10.

[4] M. auf de Heide Friedhelm, S. Christian, V. Klaus, and G. Matthias, "Energy, congestion and dilation in radio networks," in *Proc. the 14th ACM symposium on Parallel algorithms and architectures (SPAA'02)*, Winnipeg, Manitoba, Canada, 2002, pp. 230–237.

[5] L. Hu, "Topology control for multihop packet radio networks," *IEEE Transactions on Communications*, vol. 41, no. 10, pp. 1474–81, 1993.

[6] M. Burkhart, P. von Rickenbach, R. Wattenhofer, and A. Zollinger, "Does topology control reduce interference?" in *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'04)*, Tokyo, Japan, May 2004, pp. 9–19.

[7] B. Das and V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets," in *Proc. IEEE International Conference on Communications (ICC'97)*, Montreal, Canada, 1997, pp. 376–380.

[8] B. Das, R. Sivakumar, and V. Bharghavan, "Routing in ad hoc networks using a virtual backbone," in *Proc. IEEE International Conference on Computer Communications and Networks (IC3N'97)*, Las Vegas, 1997, pp. 1–20.

[9] B. Das, R. Sivakumar, and Vaduvur, "Routing in ad hoc networks using a spine," in *Proc. IEEE International Conference on Computers and Communications Networks (IC3N'97)*, Las Vegas, NV., 1997, p. 34.

[10] C. Yu-Liang and H. Ching-Chi, "Routing in wireless/mobile ad-hoc networks via dynamic group construction," *Mobile Networks and Applications*, vol. 5, no. 1, pp. 27–37, 2000.

[11] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *Proc. International workshop on Discrete algorithms and methods for mobile computing and communications (DIALM'99)*, Seattle, Washington, USA, 1999, pp. 7–14.

[12] R. Sivakumar, P. Sinha, and V. Bharghavan, "Cedar: a core-extraction distributed ad hoc routing algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1454–1465, 1999.

[13] F. Dai and J. Wu, "An extended localized algorithm for connected dominating set formation in ad hoc wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 10, pp. 908–920, 2004.

[14] J. Bolla and D. Huynh, "Adapting connected d-hop dominating sets to topology changes in wireless ad hoc networks," in *Proc. IEEE International Performance Computing and Communications Conference (PCC'06)*, Phoenix, AZ, USA, 2006, pp. 207–214.

[15] K. Sakai, M. T. Sun, and W. S. Ku, "Message-efficient cds construction in manets," in *Proc. IEEE International Symposium on Wireless Pervasive Computing (ISWPC'10)*, Modena, Italy, May 2010.

[16] A. K. M., W. Peng-Jun, and O. Frieder, "Message-optimal connected dominating sets in mobile ad hoc networks," in *Proc. the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'02)*, Lausanne, Switzerland, 2002, pp. 157–164.

[17] B. Lichun and G.-L.-A. J. J., "Topology management in ad hoc networks," in *Proc. the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'02)*, Annapolis, Maryland, USA, 2003, pp. 129–140.

[18] H. Y. Yang, C. H. Lin, and M. J. Tsai, "Distributed algorithm for efficient construction and maintenance of connected k-hop dominating sets in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 4, pp. 444–457, 2008.

[19] J. Wu, F. Dai, and S. Yang, "Iterative local solutions for connected dominating set in ad hoc wireless networks," *IEEE Transactions on Computers*, vol. 57, no. 5, pp. 702–715, 2008.

[20] M. Couture, M. Barbeau, P. Bose, and E. Kranakis, "Incremental construction of k-dominating sets in wireless sensor networks," *Ad Hoc Sensor Wireless Networks*, vol. 5, no. 1-2, pp. 47–68, 2008.

[21] Y. Li, D. Kim, F. Zou, and D.-Z. Du, "Constructing connected dominating sets with bounded diameters in wireless networks," in *Proc. International Conference on Wireless Algorithms, Systems and Applications, 2007. (WASA'2007)*, Chicago, IL, 2007, pp. 89–94.

[22] D. Kim, Y. Wu, Y. Li, F. Zou, and D.-Z. Du, "Constructing minimum connected dominating sets with bounded diameters in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 2, pp. 147–157, 2009.

[23] S. M. Aurélio and J. J. Garcia-Luna-Aceves, "Bounded-distance multi-clusterhead formation in wireless ad hoc networks," *Ad Hoc Networks*, vol. 5, no. 4, pp. 504–530, 2007.

[24] V. S. Anitha and M. P. Sebastian, "Scenario-based diameter-bounded algorithm for cluster creation and management in mobile ad hoc networks," in *Proc. IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT'09)*, Singapore, 2009, pp. 97–104.

[25] M. Q. Rieck, S. Pai, and S. Dhar, "Distributed routing algorithms for multi-hop ad hoc networks using d-hop connected d-dominating sets," *Computer Networks*, vol. 47, no. 6, pp. 785–799, 2005.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)