Paper

Load-balanced route discovery for mobile ad hoc networks

Mehran Abolhasan, Justin Lipman, and Tadeusz A. Wysocki

Abstract— This paper presents flow-aware routing protocol (FARP), a new routing strategy designed to improve load balancing and scalability in mobile ad hoc networks. FARP is a hop-by-hop routing protocol, which introduces a flow-aware route discovery strategy to reduce the number of control overheads propagating through the network and distributes the flow of data through least congested nodes to balance the network traffic. FARP was implemented in GloMoSim and compared with AODV. To investigate the load distribution capability of FARP new performance metrics were introduced to measure the data packet flow distribution capability of the each routing protocol. The simulation results obtained illustrate that FARP achieves high levels of throughput, reduces the level of control overheads during route discovery and distributes the network load more evenly between nodes when compared to AODV. This paper also describes a number of alternative strategies and improvements for the FARP.

Keywords— ad hoc routing, MANET, load-balancing, on-demand routing, protocols.

1. Introduction

Following the success of 2nd generation mobile (cellular) telephones in the late 1990's, the demand for wireless communication has continued to grow. Part of this success has been due to the growing demand in Internet type application over the wireless medium. This demand has partly been addressed through the introduction of 2.5G GPRS and more recently the 3G (WCDMA1x) networks. Other solutions becoming widely popular are wireless local area networks (also known as Wi-Fi hotspots). Such networks are designed to extend the coverage of wired networks by providing network access to mobile users. One shortcoming of the above technologies is their inability to provide a networking solution in environments where a networking infrastructure does not exists. Currently, infrastructured networks such as 2.5G, 3G and Wi-Fi hotspots exist mainly in metropolitan areas, where consumer demand is high. To address this shortcoming a networking technology is required, which can be easily and cost effectively configured without the need for a pre-existing infrastructure. One such solution is ad hoc networking. In ad hoc networks each end-user node is capable of sending, receiving and routing data packets in a distributed manner. Moreover, such networks can be configured to allow for mobility and perform routing over multiple hops. Such networks are commonly reffered to as mobile ad hoc networks (or MANETs). MANETs are still in their early development stage with the current areas of research spanning across all the levels of the traditional TCP/IP networking model. One interesting area of research in such networks is routing. Designing an efficient routing protocol for MANETs is a non-trivial task. This is primarily due to the dynamic nature of these networks, which requires intelligent strategies that can determine routes with minimum amount of overheads to ensure high levels of scalability. Consequently, researchers have proposed many different types of routing protocols for MANETs. These protocols can be categorised into three groups: proactive, reactive and hybrid routing.

Proactive routing was the first attempt at designing routing protocols for MANETs. The early generation proactive protocols such as DSDV [13] and GSR [4] were based on the traditional distance vector and link state algorithm, which were originally proposed for wired networks. These protocols periodically maintain routes to all nodes with in the network. The disadvantage of these strategies were the lack of their scalability due to exceedingly large amount of overhead they produced. More recent attempts at reducing control overhead in proactive routing can be seen in protocols such as OLSR [7] and TBRPF [3]. These protocols attempt to reduce the control by reducing the number of re-broadcasting nodes in the network.

Reactive (or on-demand) routing protocols attempt to reduce the amount of control overhead disseminated in the network by determining routes to a destination when it is required. This is usually achieved through a two phase route discovery process initiated by a source nodes. The first phase of route discovery starts by the propagation of route request (RREQ) packets through the network. The second phase is initiated when a RREQ packet reaches a node, which has a route to the destination or the destination itself, in which case a route reply (RREP) packet is generated and transimited back to the source node. When the number of flows in the network is low, reactive routing protocols produce significantly lower amount of routing overhead compared to proactive routing protocols. However, for large number of flows reactive protocols experience a significant drop in data throughput. This is because routing control packets are usually flooded (globally) throughout the entire network to find a route to the destination. To reduce the global flooding in the network a number of different strategies have been proposed. In LAR [8] and RDMAR [2] the protocols attempt to use prior location knowledge of the destination to reduce the search zone during route discovery. In LPAR [1] a combination of prior location knowledge and unicasting is used to reduce the number of re-broadcasting nodes within a search zone. In AODV [5] the source nodes use expanding ring search (ERS) to search

nearby nodes first. Therefore, reducing the number of globally propagating control packets.

Hybrid routing protocols combine both reactive and proactive routing characteristics to achieve high levels of scalability. Generally, in hybrid routing protocols, proactive routing is used within a limited region. These regions can be a cluster, a tree or a zone, which may contain a number of end-user nodes. Reactive routing is used to determine routes, which do not lie within a source node's local region. The idea behind this approach to routing is to allow nearby nodes to collaborate and reduce the number of rebroadcasting nodes. Therefore, during a route discovery only a selected group of nodes within the entire network may rebroadcast packets.

While a great deal of attention has been paid to reducing routing overhead, not much attention has been paid in ensuring a fair distribution of traffic flow (or load) between the nodes. Most routing protocols proposed for MANETs select routes based on the shortest-path which is determined using hop count as the route selection metric. This can lead to congestion or the creation of traffic bottlenecks in the network, which can results in higher levels of packets being dropped in the network and rapid depletion of resources in specific nodes. Previous work in designing better load distribution within ad hoc networks includes [6, 10, 15]. These strategies use routing load as the primary route selection criterion. In [11], the author argues that better load distribution can be achieved by flowing data over multiple routes instead of using a single route. In [14], a combination of a delay metric and hop count is used to select routes during the route discovery phase. In this paper, we propose Flow-aware routing protocol (FARP), a routing strategy which aims to reduce the amount of control overhead while ensuring a better distribution of traffic between the nodes. In FARP, a utility metric is introduced to restrict the propagation of route request packet over nodes with minimum number of active data flows from different source nodes. Therefore, congestion or the creation of bottleneck nodes is reduced.

The rest of this paper is organised as follows. In Section 2, we describe FARP. Section 3 illustrates the simulation environment, parameters and metrics used to investigate the performance of FARP with a number of routing protocols. Section 4 presents a discussion of the simulation results. Section 5 points a number of alternative strategies and improvements for FARP and Section 6 gives the conclusions of the paper.

2. Flow-aware routing protocol

The FARP employs the hop-by-hop routing strategy used in AODV. However, unlike AODV, FARP attempts to reduce the amount of control overhead while ensuring a better distribution of data traffic. This is achieved by introducing a flow-aware route discovery strategy, which selects the nodes with the least number of traffic flows.

In FARP, each node maintains a flow table, which stores a $Flow_{ID}$, a flow counter ($Flow_c$) and the ID of the previous node from which the data are received (B_{ID}). The $Flow_{ID}$ is the concatenation of the source, destination ID's of a particular flow and the node of the previous hop, which has forwarded the packet (i.e., $Flow_{ID} = S_{ID}|B_{ID}|D_{ID}$). This strategy allows each node to independently assign the unique flow IDs and identify all data flows travelling through or originating from them. The $Flow_c$ stores the number of different unique data flows that pass through each node. This includes the data flow in which the nodes act as an intermediate node and the data flows that they initiated. Note that the data flow tables maintain information about flows, which are considered as active. To do this, each node updates its data flow counter periodically using timeouts and also reactively when a broken link is reported. Similarly, new flows are added reactively, when a nodes initiates or forwards a data packet which is recorded in the flow table. The following algorithms illustrate the flow-add (FA) algorithm.

Algorithm FA
(∗ The flow-add algorithm ∗)
1. $Flow_t \leftarrow$ flow expiration time
2. $Flow_{ID} \leftarrow$ flow ID for the data packet
3. $Flow_T \leftarrow$ flow table
4. $Flow_c \leftarrow$ flow counter
5. $Flow_A \leftarrow$ flow update flag
6. $S_{ID} \leftarrow$ source node ID
7. $D_{ID} \leftarrow$ destination node ID
8. $B_{ID} \leftarrow$ previous forwarding node ID
9. $Flow_{ID} = S_{ID}|B_{ID}|D_{ID}$
10. $Found \leftarrow$ false a flag used to find flow ID
11. **for** $i \leftarrow 0$, $i < Flow_c$, $i++$
12. **if** $Flow_T[i].Flow_{ID} = Flow_{ID}$
13. $Found \leftarrow True$
14. break
15. **if** $Found = True$
16. Set($Flow_T[i].Flow_t$)
17. **else**
18. $Flow_T[i].Flow_{ID} \leftarrow Flow_{ID}$
19. $Flow_T[i].B_{ID} \leftarrow B_{ID}$
20. Set($Flow_T[i+1].Flow_t$)
21. $Flow_c++$
22. **if** $Flow_c \geq 1$ & $Flow_A != Active$
23. $Flow_A \leftarrow Active$
24. Activate the flow-delete-proactive function

In the FA algorithm, when a node has received or has initiated a data packet, it checks to see if a corresponding $Flow_{ID}$ already exists for that particular flow. If yes, it refreshes the $Flow_t$ for that flow. Otherwise, a new $Flow_{ID}$ is created and a new $Flow_t$ is set. Note that the $Flow_t$ is set by adding the current time by a timeout value[1]. More-

[1]The timeout value can be a constant or a it can be calculated dynamically from the rate at which data packets are received from a particular source.

over, the FA algorithm activates (or re-activates) the flow-delete-proactive (FD_P) function if there are one or more entries in the flow table.

The following algorithms illustrate the FD_P and flow-delete-reactive (FD_R) strategies, respectively.

Algorithm FD_P
(∗ The flow-delete-proactive algorithm ∗)
1. $Time_c \leftarrow$ current time
2. $Flow_T \leftarrow$ the flow table
3. $Flow_c \leftarrow$ flow counter
4. $Flow_t \leftarrow$ flow expiration time
5. $Flow_A \leftarrow$ flow update flag
6. $Total_{Flows} \leftarrow Flow_c$
7. **while** $(Flow_c > 0)$
8. **for** $i \leftarrow 0$, $i < Total_{Flows}$, $i++$
9. **if** $Flow_T[i].Flow_t > Time_c$
10. delete $Flow_T[i]$
11. $Flow_c --$
12. **if** $Flow_c = 0$
13. $Flow_A \leftarrow InActive$

Algorithm FD_R
(∗ The flow-delete-reactive algorithm ∗)
1. $Flow_T \leftarrow$ flow table
2. $B_{ID} \leftarrow$ intermediate node ID in the broken link
3. $Flow_c \leftarrow$ flow counter
4. $Total_{Flows} \leftarrow Flow_c$
5. **for** $i \leftarrow 0$, $i < Total_{Flows}$, $i++$
6. **if** $Flow_T[i].B_{ID} = B_{ID}$
7. delete $Flow_T[i]$
8. $Flow_c --$
9. **if** $Flow_c = 0$
10. $Flow_A \leftarrow InActive$

The FD_P algorithm is used to periodically scan the flow table for expired $Flow_{ID}$s. This is achieved by comparing the flow expiration time (i.e., $Flow_t$) for each $Flow_{ID}$ with the current time. If the $Flow_t$ is greater than $Time_c$, then the flow entries for that particular flow is removed and the $Flow_c$ is decremented. Note that the FD_P function will be deactivated when the $Flow_c$ is set to zero (i.e., when the flow table is empty).

The FD_R algorithm is used to remove flow ID's of the data packets travelling over links which have become inactive. The invalid flow IDs are removed by comparing the ID of the broken link with the ID of the forwarding node (previous hop), then removing the entries in the flow table, which are associated with the broken link. Each time a route entry table is removed, the $Flow_c$ is also decremented. When the flow table scanning phase has been completed, if the flow counter has been set to zero, the flow update flag is set to inactive. This is done to deactive the FD_P function.

When a node has data to send and route to the required destination is not available, then route discovery is initiated. The flow-aware route discovery algorithm is outlined below[2].

Algorithm FSF
(∗ The flow-based selective flooding algorithm ∗)
1. $RREQ_{max} \leftarrow$ maximum number of route request retries
2. $Flow_\tau \leftarrow \tau$ data flow packet threshold
3. $Flow_F \leftarrow$ flow metric
4. $Flow_N \leftarrow 0$ (∗ no metric to be used ∗)
5. $P \leftarrow \{0.125, 0.25, 0.5, 0.75, 1.0\}$ (∗ maximum % of data flow allowed ∗)
6. $RREQ_{max} \leftarrow 4$
7. **for** $i \leftarrow 0$, $i \neq RREQ_{max}$, $i++$
8. $Flow_F \leftarrow Flow_\tau.P_i$
9. **if** $Flow_F = 0$
10. $Flow_F \leftarrow 1$
11. forward_RREQ($Flow_F$)
12. wait for reply
13. **if** $Route = found$
14. break loop
15. initiate data transmission
16. **if** $Route = not\,found$
17. Forward_RREQ($Flow_N$)
18. wait for reply
19. **if** $Route = found$
20. initiate data transmission
21. **else**
22. return route not found

In the FSF algorithm, the source node begins calculating a flow metric ($Flow_F$), which states the maximum number of flows allowed for each node to be able to re-broadcast the RREQ packet. Therefore, each node only re-broadcast a RREQ packet if the number of flows it handles is less than the number speficied in $Flow_F$ (i.e., when $flow_c < Flow_F$). In the FSF algorithm five different levels of data flow (i.e., P) can be selected to calculate the flow metric. During each route request retry, this value is increased until $i = RREQ_{max}$. If the route to the destination is still not found, then source node transmits a RREQ without a flow metric (i.e., $Flow_N$), which allows all intermediate nodes to rebroadcast. If the source node determines more than one route to the required destination, it uses the one with the lowest number of flows and the shortest path. Furthermore, if two routes are found with identical number of flows and hops (which have also least number of flows and hops), then the preferred route is randomly selected.

When a source node has data to send, and a fresh (or active) route already exists or has been determined through a route discovery, then a $Flow_{ID}$ is created and stored, and the data is forwarded to the next hop. Each forwarding node then creates their own flow IDs (as described previously) and continue forwarding the data packets. This process continues (including at the destination node) until the destination node is reached. Furthermore, each consequtive data packet

[2]We refer to this algorithm as flow-based selective flooding (FSF).

is used to update the lifetime of each flow ID (if the flow ID already exists).

To illustrate how FSF algorithm works, assume that $Flow_\tau = 1$ and S1, S2 and S3 (see Fig. 1) want to send data to D1, D2 and D3. Using shortest path (SP) routing,

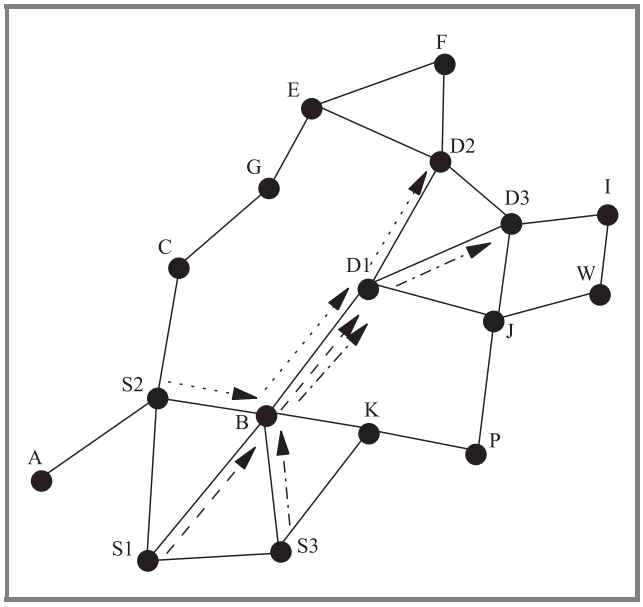

Fig. 1. Data packet flow using SP routing only.

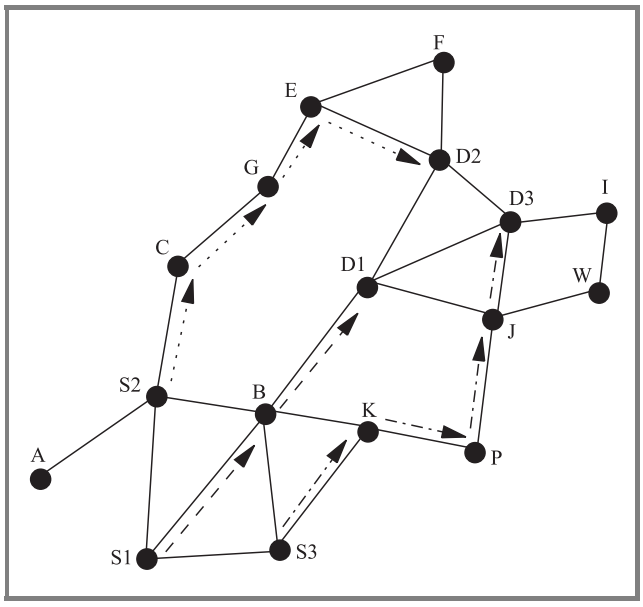

Fig. 2. Data packet flow using FSF.

all data packets travel through node B and D1. Thus creating possible performance bottlenecks at these nodes. In FSF (Fig. 2), the route discovery strategy uses a combination of data flows restriction and SP routing to distribute the packets through nodes C, B and K, instead of through node B only (as was the case in Fig. 1). As a result, FARP ensures a better distribution of data traffic than using purely SP routing.

To illustrate how FARP can reduce the number of control packets, let us assume that S (Fig. 3) wants to send data to D. In this scenario, under SP routing the route discovery

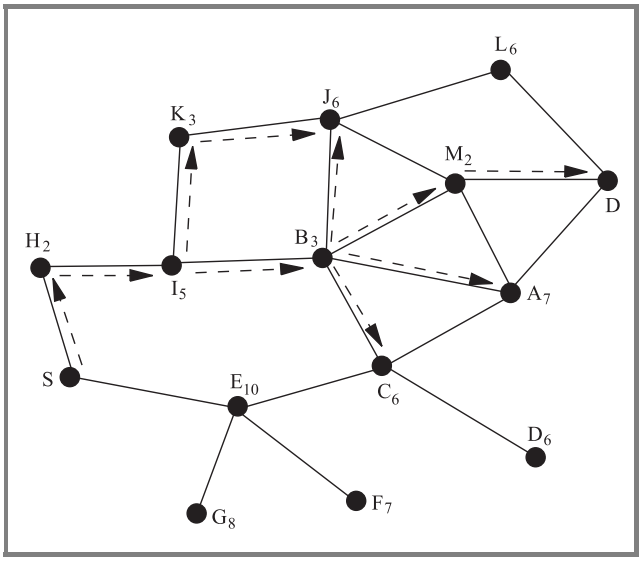

Fig. 3. Illustration of control overhead reduction in FARP (note in X_Z, X represents the node ID and Z is the number of flows).

phase results in transmission of 15 RREQ packets (i.e., all nodes broadcast). However, in FARP, only 6 nodes broadcast the RREQ packet. Thus, a control overhead reduction of 60% is achieved. In scenarios where the number of nodes and traffic level is high, it is expected that FARP will experience significant drop in the number of control packets when compared to other SP-based on-demand routing protocols such as AODV. In Section 4, FARP is compared with AODV using simulation studies performed over densely populated mobile ad hoc network, with multiple number of flows.

3. Simulation model

This section describes the scenarios and parameters used in simulation studies performed for FARP. It also illustrates the performance metrics used to compare FARP with AODV.

3.1. Simulation environment and scenarios

The GloMoSim [9] simulation package was chosen to run the simulations. GloMoSim is an event driven simulation tool designed to carry out large simulations for mobile ad hoc networks. The simulations were performed for 10, 20 and 100 node networks, migrating in a 1000 m $\times$ 1000 m area. IEEE 802.11 DSSS (direct sequence spread spectrum) was used with maximum transmission power of 15 dbm at a 2 Mb/s data rate. In the MAC layer, IEEE 802.11 was used in DCF mode. The radio capture effects were also taken into account. Two-ray path loss characteristics was considered as the propagation model. The antenna height

was set to 1.5 m, the radio receiver threshold was set to –81 dbm and the receiver sensitivity was set to –91 dbm according to the Lucent wavelan card [12]. Random waypoint mobility model was used with the node mobility ranging from 0 to 20 m/s and pause time was set to 0 s for continuous mobility. The simulations ran for 200 s (we kept the simulation time lower due to a very high execution time required for the 40 flow scenario) and each simulation was averaged over eight different simulation runs using different seed values.

Constant bit rate (CBR) traffic was used to establish communication between nodes. Each CBR packet contained 512 bytes and each packet were transmitted at 0.25 s intervals. The simulation was run for 5, 10, 20 and 40 different client/server pairs[3] and each session began at a randomly selected time and was set to last for the duration of the simulation.

Table 1
FARP simulation parameters

Parameters	Values
Flow timeout	3 s
Flow expiration time	2 s
Flow threshold	8
RREQ retry times	6

The FARP routing protocols was implemented on the top of the AODV algorithm. Table 1 illustrates the simulation parameters used for FARP. Note that the flow timeout represents the timeout interval at which the flow table entries are updated. The flow expiration time represents the lifetime of each flow. The flow threshold is used to assume a maximum number of flows at each node. This is used in the FSF algorithm. The RREQ retry times represents the number of times a source can initiate a route discovery before the destination is seen as unreachable.

3.2. Performance metrics

The performance of each routing protocol is compared using the following performance metrics:

- packet delivery ratio (PDR),
- control packet overhead (O/H),
- end-to-end delay,
- total flows per node (TFN).

The PDR is the ratio of the number of number of packets received by the destination to the number of packets sent by the source. Control packet overhead presents the number of control packets transmitted through the network. The end-to-end delay represents the average delay experienced by each packet when travelling from the source to the destination. The TFN represents the total number of data flows handled by each node in the network for the complete duration of the simulation. The above metrics where taken for different values of pause time.

[3]Note that the terms client/server, src/dest and flows are used interchangeably.

4. Results

This section presents the simulation results obtained for FARP and AODV. A performance comparison between both protocols is also provided.

4.1. Packet delivery ratio

Figures 4 and 5 illustrate the PDR results obtained for the 20 and 100 node scenarios. These figures illustrate the packet delivery performance of AODV and FARP in a small to medium size mobile ad hoc network. In the 20 node scenarios both FARP and AODV achieve over 98% PDR. However, in the 100 node scenario it can be seen that FARP achieves a higher level of packet delivery than AODV when node mobility is high (i.e., for short pause times). This is because FARP reduces the probability of establishing

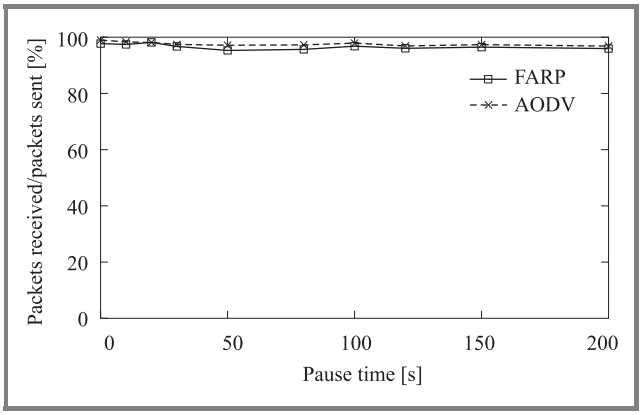

Fig. 4. Packet delivery ratio versus pause time: 20 nodes and 10 flows.

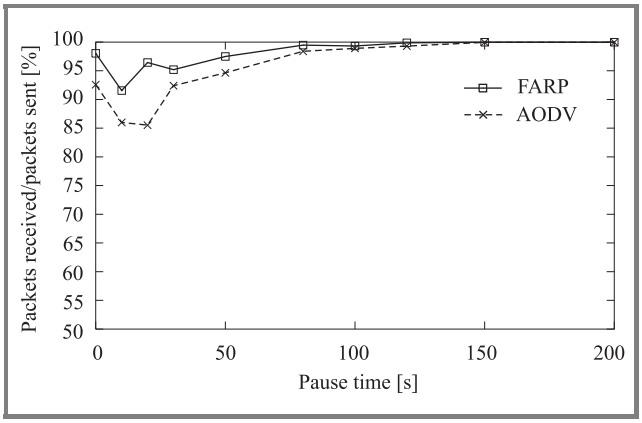

Fig. 5. Packet delivery ratio versus pause time: 100 nodes and 50 flows.

routes over bottleneck (or saturated nodes). Thus in FARP, data packets have a better chance of reaching the required destination than in AODV. Furthermore, FARP introduces a more self-selective approach to flooding than AODV. This means that not every node in the network need rebroadcast control packets. Hence, there is often reduction in channel contention between nodes and smaller chance of packets being lost due to interference and buffer overflows when compared to the blind flooding approach employed in AODV.

4.2. Control packets overhead

Figures 6 and 7 illustrate the number of control packets introduced into the network for the 20 and 100 node scenarios, respectively. In both scenarios it can be seen that FARP produces fewer control packets than AODV. This is

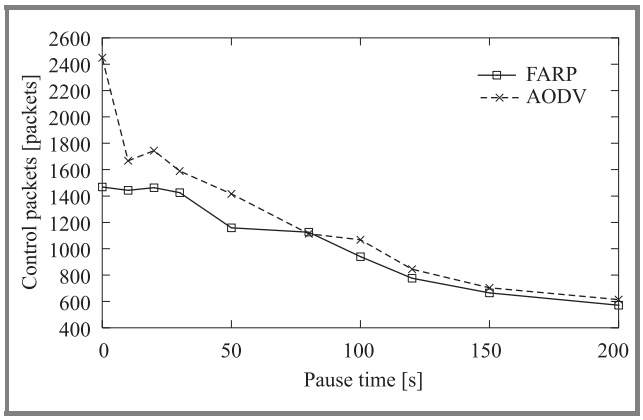

Fig. 6. Control packet overhead versus pause time: 20 nodes and 10 flows.

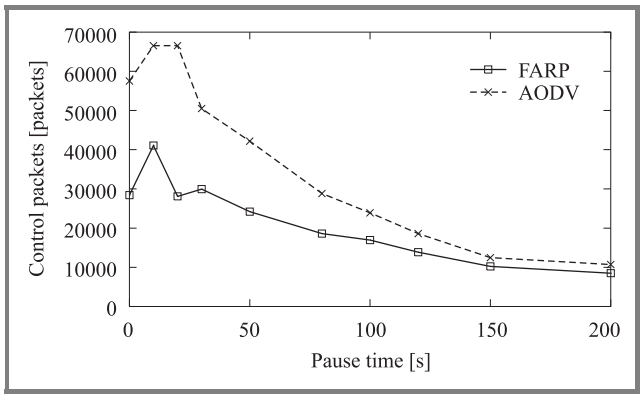

Fig. 7. Control packet overhead versus pause time: 100 nodes and 50 flows.

more evident when mobility is high, because in high mobility both protocols initiate more route discoveries due to more frequent route failures. However, in FARP each route discovery may result in fewer number of control packet rebroadcasts than AODV, due to restriction of flooding over nodes which have fewer flows thereby reducing the number of rebroadcasting nodes when compared with AODV.

4.3. End-to-end delays

Figures 8 and 9 illustrate the end-to-end delay introduced for the 20 and 100 node network scenarios, respectively. In the 20 node scenario, both AODV and FARP produce similar levels of end-to-end delay. This is because

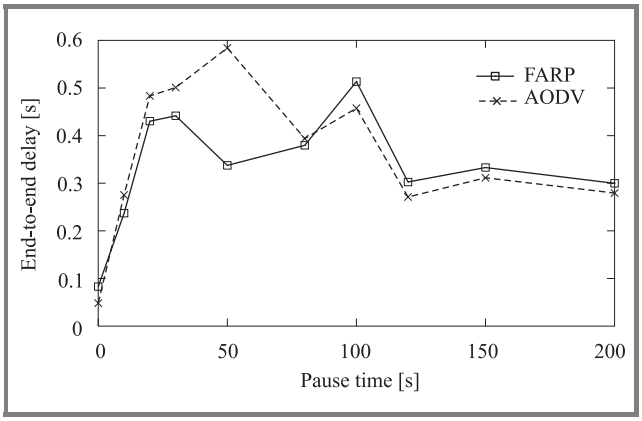

Fig. 8. End-to-end delays versus pause time: 20 nodes and 10 flows.

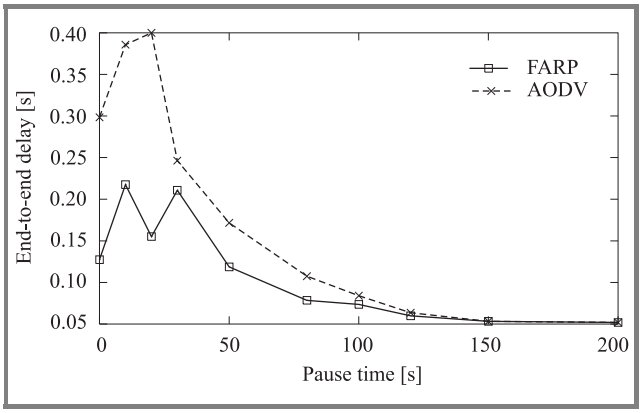

Fig. 9. End-to-end delays versus pause time: 100 nodes and 50 flows.

the amount of traffic introduced into the network is lower than the available bandwidth and the capacity of each node (i.e., no long queue at each node). In the 100 node network with 50 flows, FARP achieves significantly lower end-to-end delay than AODV when mobility is high. This is because AODV produces significantly more control overhead than FARP (as described previously in the control packet overhead results), which increases channel contention between nodes and may increase the time that each data packet spends in buffers before being transmitted.

4.4. Flow distribution

Figures 10, 11 and 12 illustrate the number of different flows handled by each node for zero pause time (i.e., constant node mobility) for the entire duration of the simulation. In the 10 node and 20 node scenario, FARP produces

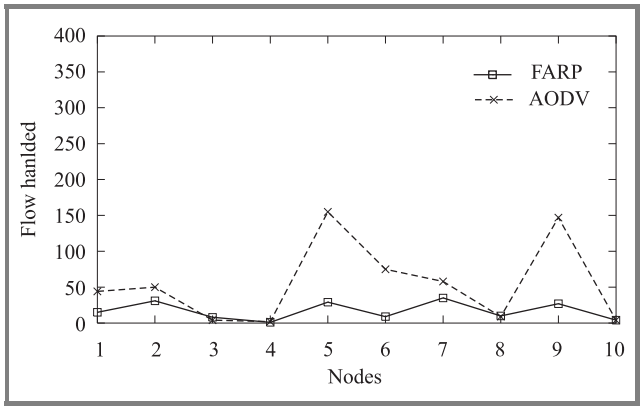

Fig. 10. Flow distribution: 10 nodes and 5 flows.

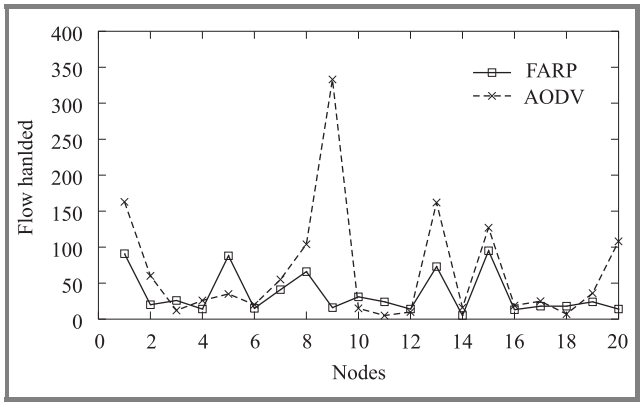

Fig. 11. Flow distribution: 20 nodes and 10 flows.

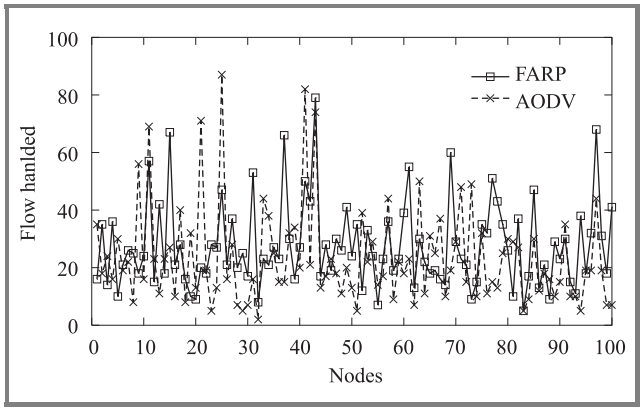

Fig. 12. Flow distribution: 100 nodes and 50 flows.

significantly better flow distribution than AODV. This can be seen by the flatness of the curves. In FARP, the total number of flows at each node varies between 10 to 40 for the 10 node scenario, and 10 to 90 flows for the 20 node scenario. However, in AODV the flows vary between 0 to 150 flows for the 10 node scenario and 0 to 340 flows for the 20 node scenario. Hence, there are larger spikes in the AODV graph than in FARP. This indicates that in FARP flows are more evenly distributed than AODV. In the 100 node scenario, the flow distributions achieved in AODV and FARP are more closely matched than the other less dense scenarios. This is because each node has a higher probability of handling data packets due to the larger traffic density. However, with close observation of the 100 node graph it can be seen that AODV still experiences larger variation in flow distribution. For example, the smallest flow count experienced by a node in AODV is close to 0 flows and the largest is around 90 flows, whereas in FARP the smallest value is close to 8 flows and the largest is close to 78 flows.

5. Alternative strategies and improvements

5.1. Dynamic flow threshold selection

In the FSF algorithm, the flow threshold (the limit for the number of flows allowed at each node) was chosen as a simulation parameter. Therefore, each node in our simulations used a static value for the flow threshold. The disadvantage of a static flow threshold is that it may not always allow for the best flow distribution in the network. To make more accurate prediction of flow limits and better flows distribution, each node must make these decisions dynamically based on the current conditions of the network. One way to calculate the flow threshold dynamically is through the use and exchange of neighbour flow information. In this strategy, each node exchange flow information with their neighbouring nodes (using hello packets) and calculates an average flow per neighbour and the maximum number of flows, which can be experienced by each node at each particular region. Using this information the first few RREQ propagation can be restricted only to the nodes that are handling average or lower levels of flows.

5.2. Rate adaptive flow timeout selection

In our FARP simulations, the flows that are not refreshed every 2 s or less are deleted from the flow table. The disadvantage of this is that different applications may be transmitting data at different rates. Therefore, by assigning a static flow timeout, the flow table may be storing each flow ID for a longer or shorter time than it is required. To overcome this, the flow timeout value can be set by observing the rate at which data packets arrive at each node and assigning a timeout value, which closely matches the expected arrival time.

6. Conclusions

In this paper, we introduced a new routing strategy for mobile ad hoc networks. This routing strategy is referred to as flow aware routing protocol. In FARP, a new route discovery strategy is introduced, which uses the flow information kept at each node to reduce the number of control packets

while ensuring better distribution of data packets between the nodes in the network. This is achieved by restricting the RREQ retransmission over nodes that have the lowest number of flows. We implemented FARP on the top of AODV and compared the performance through simulation. Our results show that FARP reduces the number of control packets transmitted through the network, while achieving improved data flow distribution in the network. In the future, we plan to investigate the performance of FARP over large networks with high levels of mobility.

References

[1] M. Abolhasan, T. A. Wysocki, and E. Dutkiewicz, "LPAR: an adaptive routing strategy for MANETs", *J. Telecommun. Inform. Technol.*, no. 2, pp. 28–37, 2003.

[2] G. Aggelou and R. Tafazolli, "RDMAR: a bandwidth-efficient routing protocol for mobile ad hoc networks", in *ACM Int. Worksh. Wirel. Mob. Multimed. WoWMoM*, Seattle, USA, 1999, pp. 26–33.

[3] B. Bellur, R. G. Ogier, and F. L. Templin, "Topology broadcast based on reverse-path forwarding routing protocol (tbrpf)", Internet Draft, 2003, draft-ietf-manet-tbrpf-06.txt

[4] T.-W. Chen and M. Gerla, "Global state routing: a new routing scheme for ad hoc wireless networks", in *Proc. IEEE ICC*, Atlanta, USA, 1998.

[5] S. Das, C. Perkins, and E. Royer, "Ad hoc on demand distance vector (AODV) routing", Internet Draft, 2002, draft-ietf-manet-aodv-11.txt

[6] H. Hassanein and A. Zhou, "Routing with load balancing in wireless ad hoc networks", in *Proc. ACM MSWiM*, Rome, Italy, 2001.

[7] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks", in *Proc. IEEE INMIC*, Orlando, USA, 2001.

[8] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks", in *Proc. Fourth Ann. ACM/IEEE Int. Conf. Mob. Comput. Netw. Mobicom'98*, Dallas, USA, 1998.

[9] UCLA Parallel Computing Laboratory and Wireless Adaptive Mobility Laboratory, "Glomosim scalable simulation environment for wireless and wired network systems", 2003, http://pcl.cs.ucla.edu/projects/glomosim/

[10] S. J. Lee and M. Gerla, "Dynamic load-aware routing in ad hoc networks", in *Proc. ICC 2001*, Helsinki, Finland, 2001.

[11] S. J. Lee and M. Gerla, "SMR: split multipath routing with maximally disjoint paths in ad hoc networks", in *Proc. ICC 2001*, Helsinki, Finland, 2001.

[12] Lucent, "Orinoco PC card", 2003, http://www.lucent.com/orinoco

[13] C. E. Perkins and T. J. Watson, "Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers", in *ACM SIGCOMM'94 Conf. Commun. Archit.*, London, UK, 1994.

[14] J. H. Song, V. Wong, and V. Leung, "Load-aware on-demand routing (laor) protocol for mobile ad hoc networks", in *IEEE Veh. Technol. Conf. VTC-Spring*, Jeju, Korea, 2003, pp. 1753–1757.

[15] K. Wu and J. Harms, "Load-sensative routing for mobile ad hoc networks", in *Proc. IEEE ICCCN'01*, Scottsdale, USA, 2001.

Mehran Abolhasan received the B.E. in computer engineering with honours from the University of Wollongong in 1999. He completed his Ph.D. in the School of Computer, Electrical and Telecommunications Engineering, University Wollongong in 2003. In July 2003, he started working as a Research Fellow with Smart Internet Technology CRC and Office of Information and Communication Technology (OICT) within the Department of Commerce in NSW. In July 2004, he joined the desert knowledge CRC and Telecommunication and IT Research Institute. His research interests are wireless and ad hoc network scalability, medium access control (MAC), unicast and multicast routing protocols and QoS. Doctor Abolhasan has authored several research publications in this area, and is currently a Member of IEEE.
e-mail: mehran@titr.uow.edu.au
Telecommunications and IT Research Institute (TITR)
University of Wollongong
Northfields Ave
Wollongong, NSW 2522, Australia

Justin Lipman received a B.E. in computer engineering and Ph.D. in telecommunications engineering from the University of Wollongong in 1999 and 2004, respectively. In 2003 he was employed as a Research Fellow with the Smart Internet Technology Cooperative Research Centre (CRC-SIT) working on the Nimity Project. In 2004, he moved to Alcatel Shanghai Bell's Research and Innovation Centre in Shanghai, China, as a project manager to lead an innovating team looking at future wireless communications. Doctor Lipman's research interests are diverse but focus in general upon mesh/ad hoc networks, 3G/B3G/4G networks, sensor networks and distributed systems.
e-mail: justin@titr.uow.edu.au
Telecommunications and IT Research Institute (TITR)
University of Wollongong
Northfields Ave
Wollongong, NSW 2522, Australia

Tadeusz A. Wysocki – for biography, see this issue, p. 23.