International Journal of Future Generation Communication and Networking
Vol.7, No.4 (2014), pp.103-118
http://dx.doi.org/10.14257/ijfgcn.2014.7.4.10

A New Approach to Estimate RED Parameters Using Function Regression

Wentao Wang, Hao Wang, Wan Tang and Feng Guo

College of Computer Science, South-Central University for Nationalities, No.182 Minyuan Road, Wuhan 430074, China
wangwt@mail.scuec.edu.cn, haowang354@163.com

Abstract

Random Early Drop (RED) is widely applied in Mobile Ad-hoc NETworks (MANETs) for congestion control. It randomly drops packets to prevent congestion from occurring, while keeping proper queue size of Route Request (RREQ) packets at the same time. Unfortunately, in the case of severe congestion and large amounts of RREQ packets in the queue, RED algorithm with tune parameters is not able to improve the congestion status properly. This paper proposes a Dynamic RED (DRED) mechanism based on the fitting curve of packet delivery ratio and packet sending rate, which can be readily achieved in practice. DRED presupposes threshold value for initiating and dynamically changes it according to the function regression. Compared to First In First Out (FIFO) and RED with tune parameters mechanisms, simulation results show that DRED has better performance in terms of average end-to-end delay, Hello packet overhead and packet delivery ratio, while avoiding obvious increase of routing discovery frequency.

Keywords: *Ad-hoc NETworks, congestion control, function regression, RED*

1. Introduction

In the last few years, performance comparisons of routing protocols in Ad-hoc networks have been deeply studied [1]. As an effective congestion control mechanism in Ad-hoc Networks, RED algorithm is arguably the most widely studied early random detection scheme. Although early random detection schemes can potentially outperform traditional drop-tail schemes in presence of congestion control, RED is often difficult to parameterize random early detection threshold values under different congestion scenarios. In addition, the effectiveness of such schemes in presence of congestion control is far less understood. Furthermore, there is a need for constant fine-tuning of parameters to adapt current network conditions. To that end and based on simplified models, guidelines have been proposed in [2] for setting RED parameters. However, most studies on RED are based on heuristics or simulations rather than a systematic approach. A significant research work focused on the fine tuning of the RED parameters. Literature [3] presented an analytical study aiming at addressing the loss and delay tradeoff of the RED queuing discipline in order to identify the settings of the RED parameters. Literature [4] proposed a Statistical Adapting RED to dynamically tune RED parameters based on the standard deviation of instantaneous queue size. In [5], a new congestion control algorithm，named CCAD, was presented. With the end-to-end traffic flow as its study object and adopting rate-based end-to-end congestion control strategy, CCAD tries to avoid congestion in links. In [6-10], signal strength information is given to avoid the failure of link. Their ideas are (a) if the signal strength measurements indicate that a link failure is most likely due to a neighbor moving out of range,

ISSN: 2233-7857 IJFGCN

neighbor nodes will facilitate the use of temporary higher transmission power to keep the link alive and, (b) if the measurements of signal strength indicate that a link is considered to be broken, corresponding source node will initiate a route re-discovery proactively before the link actually fails. Detecting integrity of the link according to signal strength has an obvious effect on power control and packet delivery ratio. In [11], a new stable congestion control method based on expanded Explicit Congestion Notification (ECN) mechanism was proposed to respond packet losses in wireless environment more properly. The mechanism expanded ECN uses two different bits to display the congestion level. With various busy degree of Ad-hoc Networks channel, senders can know more information about network status and destination nodes can feed back packets more accurately.

However, we have found experimentally that RREQ queue will be blocking as the communication traffic increases frequently and then a part of packets cannot arrive at the destination node. In this scenario, destination node has no feedback response to source node. Then the source node will restart the process of routing discovery, and broadcast a larger number of RREQ packets to the network, causing more repeated packets in the RREQ queue with limited size. Because of a large number of RREQ packets blocking in the queue, the source node cannot receive response from destination node, and the route to destination node will be considered unreachable. Under the circumstance, it is not necessary to send Hello packets periodically to detect integrity of the link. So we need to control the level of congestion in large traffic.

Literature [12] proposed an AODV-AH algorithm, the Hello packet interval is calculated according to the number of packets transported by neighbor nodes per unit time, which is also the standard to judge active degree of neighbor nodes. Literature [13] presented Adaptive Hello Rate (AHR), a two-state adaptive mechanism for adjusting Hello packets interval in AODV, *i.e.*, topology changes frequently and slowly. The interval of Hello packets changes based on the variation degree of topology. A Turnover based Adaptive hello Protocol (TAP) algorithm is presented in [14], to enable nodes in mobile networks to dynamically adjust the frequency of their Hello messages based on the current mobility speed of nodes. TAP uses a directed graph to describe the Hello packet when the new neighbors join in. Nodes only need to periodically make samples of their table to compute the current turnover and adapt their Hello frequency. Literature [15] proposed a Gilbet model, which uses Hello packet loss rate to discuss the integrity of the link. The model gives an efficient theoretical method to set Hello packet parameters according to the route discovery probability. A new routing protocol, called Enhanced Ad-hoc on Demand Distance Vector (E-AODV) routing protocol, is designed in [16]. E-AODV merges the Blocking Expanding Ring Search (B-ERS) & Routing packets as Hello packets techniques to reduce routing overhead.

Furthermore, [17] improved response time as well as the number of packets without affecting throughput of bottleneck link. In the modification, packets are dropped only when high congestion occurs at node. Literature [18] proposed a queue management algorithm according to the weighted fairness of flows in wireless networks. In [19], Delay Feedback Congestion Control (DFCC) was presented. DFCC reduces the packet loss ratio by adjusting link delay dynamically when congestion happens and it can ensure the stability of average length of the queue. Instead of deflecting bursts at an immediate point of contention, Random Early Deflection (RED-f) proposed in [20] triggers deflection ahead of time, and thus, offers additional routing paths and reduces the burst loss rate due to contention. Literature [21] proposed Harsh RED (HRED) to properly address slow start in time. Stable Time RED (ST-RED) stabilizes the calculation timing for average queue length leading to suppress the queue fluctuations, and improves delay and jitter performances [22]. Literature [23] proposed Linear Interpolation RED (LIRED) method to develop packet drop probability function.

According to our knowledge, there are few studies on RED with large traffic, especially on the buffer size of RREQ queue in Ad-hoc Networks. The performance of the algorithms based on RED and tune parameters are not satisfactory because of the slow response to congestion when the number of RREQ packets is large. To address this problem, in this paper, we researched the statistical data of packet delivery ratio, sending rate, route discovery frequency and Hello packet interval, and drew the fitting curve of them. Through the fitting curve we have found that the curve of packet delivery ratio and sending rate is exponential function. Therefore, by taking advantage of the curve of exponentially distributed, preset values of RED threshold can be determined before network starts.

Thus, as the process of increasing RREQ packets obeying Poisson process and the serving time is exponentially distributed in M/M/1 model, this paper proposes a congestion control mechanism called Dynamic RED (DRED) based on the fitting curve of packet delivery ratio and sending rate. DRED make preset values of responding threshold and dynamically adjust them according to the fitting curve of packet delivery ratio and sending rate.

The remainder of this paper is organized as follows. First we introduce RED algorithm in Section 2. Then we focus on DRED and its interactions with RED algorithm in Section 3. In Section 4, numerical results and theoretical analysis of both proposed and conventional models are compared and analyzed at different scene. Finally, we conclude the paper in Section 5.

2. Random Early Detection (RED)

The RED algorithm uses a weighted average of the total queue length to determine when to drop packets. When a packet arrives at the queue, if the weighted average queue length is less than a minimum threshold value q_{min}, no drop action will be taken and the packet will simply be queued. If the average queue length is greater than q_{min} but less than a maximum threshold q_{max}, an early drop test will be performed as described below. As shown in Figure 1, an average queue length, which is in the range between the two thresholds q_{min} and q_{max}, indicates some congestion has begun and flows should be notified via dropped packet. If the average is greater than the maximum threshold q_{max}, a forced drop operation will occur. An average queue length in this range indicates persistent congestion and packets must be dropped to avoid a persistently full queue. (The forced drop is also used when the queue is full but the average queue length is still below the maximum threshold.) Note that by using a weighted average, RED avoids overreaction to bursts and instead reacts to longer-term trends. Furthermore, because the thresholds are compared to the weighted average (with a typical weighting factor w_q of 1/512), it is possible that no forced drops will take place even when the instantaneous queue length is quite large. The early drop action in the RED algorithm probabilistically drops the incoming packet when the weighted average queue length is between q_{min} and q_{max}. In contrast, the forced drop action in the RED algorithm is guaranteed to drop the incoming packet. In the case of early drops, the probability of dropping the packets depends on several other parameters of the algorithm. An initial drop probability $P\_drop \leftarrow max_p (Q\_avg - q_{min}) / (q_{max} - q_{min})$ is computed, where q_{max} is the maximum drop probability (an additional control parameter) and Q_avg is the weighted average queue length. The actual drop probability is a function of the initial probability and a count of the number of packets queue since the last packet was dropped: $p_a \leftarrow p_{drop} / (1 - count \cdot p_{drop})$. Given a weighted average queue size, the impact of q_{min} depends on both q_{max} and Q_avg. This means that the appropriate q_{min} can be found for good performance, but it may only be in

combination with certain values of q_{max} and Q_avg. In principle, this is the usual practice for tuning parameters.

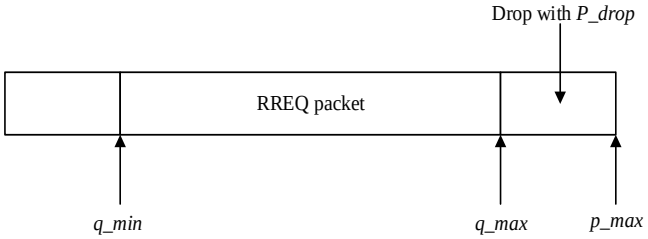

Figure 1. RREQ Packet Queue

3. Dynamic RED

Literature [4] presented an analytical study focusing on the fine tuning of RED parameters. Utilizing a statistical analysis approach, [3] formulated an optimization problem aiming at addressing the loss and delay tradeoff of RED queuing discipline. It is a two-phase iterative solution. In the first phase, fixed threshold values q_{min} and q_{max} are given. And the only decision variable Q_avg is calculated at the boundary point of a constraint function. In the second phase, q_{min} and q_{max}, as the fixed values obtained by the solution of the first phase, are the decisive variables in solving the optimization problem of dropping possibility. Therefore, they are two iterative phases. The threshold values, *i.e.*, q_{min} and q_{max}, vary according to the queue capacity K which is a fixed value. The weight w_q used to calculate thresholds is also fixed.

However, the approach mentioned above is not appropriate because the threshold values are fixed. Therefore, in this paper, we establish a novel congestion control model consisting of two parts. The first part is to set threshold values and adjust them dynamically according to the fitting curve of packet delivery ratio and sending rate, and the second part is to control the Hello packet interval. Hello packet interval and life period is regulated according to the average RREQ queue length dynamically.

3.1. Q_min and Q_max

When the traffic in network increases, we figure out the packet delivery ratio in different packet transmission rate. Then we draw the fitting curve of packet delivery ratio and sending rate, as shown in Figure 2.

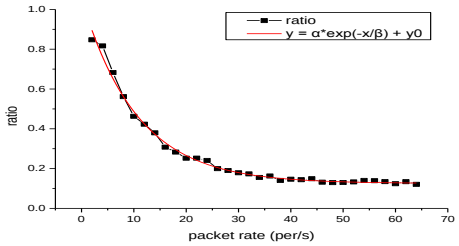

Figure 2. Fitting Curve of Packet Delivery Ratio and Sending Rate

Figure 2 depicts the packet delivery ratio in different packet transmission rate. It shows that the relation between packet delivery ratio and transmission rate approximately follows an exponential distribution. Assume that the ratio is bounded by the exponential distribution:

$$F(x) = \alpha e^{-x/\beta} + f(\lambda) \quad (1)$$

where $f(\lambda)$ is the weight to calculate ratio, and varies within a very small range. λ is the average length of RREQ packet queue (Q_avg) calculated using (2).

$$Q\_avg = \frac{\sum_{t=0}^{t} Q\_now}{t} \quad (2)$$

In (1), α and β are the parameters to fit the curve. They can be calculated by some statistical software, *e.g.,* MATLAB. As input process is a Poisson distribution in M/M/1 model, we design a distribution function to calculate weight $f(\lambda)$. The function should satisfy the following constraints with parameter λ:

(a) $f(\lambda)$ reduces as λ becomes larger, and $f(\lambda)$ will tend to 0 if λ is approaching to infinity.

(b) $f(\lambda)$ becomes larger as λ increases from initialization. $f(\lambda)$ will not increase until λ reach a certain value and then $f(\lambda)$ becomes smaller.

(c) $f(\lambda)$ is nonzero during network initialization phase.

In M/M/1 model, the number of arrivers obeys Poisson distribution, and the weight $f(\lambda)$ is the tune parameter to calculate packet delivery ratio. According to what has been discussed above, we can consider $f(\lambda)$ as a Poisson process.

$$f(\lambda) = \frac{\lambda^k}{k!} e^{-\lambda} \quad (0 \le k \le Q\_lim, 0 \le \lambda \le Q\_lim) \quad (3)$$

In formula (3), k is the current length of RREQ packet queue and $f(\lambda)$ ranges from 0 to 1. We set a confidence interval based on the ratio to calculate Q_now. Then the confidence interval of (1) is $[\chi^2_{1-\sigma/2}(2n)/2n\lambda, \chi^2_{\sigma/2}(2n)/2n\lambda]$, where n is the maximum length of RREQ queue and it is a fixed value, σ is the confidence level, and Q_lim is calculated by (4).

$$Q\_lim = \chi^2_{\sigma/2}(2n) / 2n\lambda \quad (4)$$

By the statistics of the average length of RREQ packet queue under different packet sending rate, we also obtain the relation between the two parameters, as shown in Table 1.

Table 1. Relation of Packet Rate and Q_avg

Packet rate(packet/s)	Q_avg
2	1
12	13
22	29
32	38
42	54
52	56
62	58

Since the relation between data packet transmission rate and the average length of RREQ queue is positive linear, packet transmission rate reflects the average queue length in RREQ queue. We count the average length Q_avg in the case of different data packet transmission rate, and calculate the packet delivery rate according to different average length using (1). If it is a network with tolerate delivery ratio, we can make preset values of thresholds Q_min and Q_max based on the expected delivery ratio before network starts. We can also adjust Q_min and Q_max dynamically according to Table 1 when RREQ queue is blocking. This is our methods to calculate Q_min and Q_max and improve congestion based on the fitting curve.

However, if $Q\_avg > Q\_max$, the mechanism will drop packets in RREQ queue with the possibility P_drop.

3.2. P_drop

The average length of RREQ queue is a typical "growth and consumption" process in practice, so how to control the "growth and consumption" congestion process is what we want to discuss in this paper. We simplify the process as a queuing model, and describe the congestion control model with three parameters, namely input process, serving process and queuing rules, it is an M/M/1 model.

The input process is used to describe the arrival of request packets. We assume that the packets arrive randomly. Each source node generates a request. Their time intervals all obey the same exponential distribution. The serving process mainly refers to the operation of forwarding and deleting RREQ messages. The queuing rules specify how to handle congestion message: (a) Drop packets in the queue when the packet is in waiting state. This rule is called the "clear all blocking". (b) Don't execute the delete operation. But in this case, messages have to be in a waiting state. So the order of message service in the queue will be usually specified, such as random service.

Let $\{ L(t), t \geq 0 \}$ be the length of RREQ queue at time t. Then the probability of remaining steady state in the queue can be presented by formula (5).

$$P_j = \begin{cases} \dfrac{1}{1 + \sum_{k=1}^{\infty} \left(\dfrac{\lambda_0 \lambda_1 \cdots \lambda_{k-1}}{\mu_1 \mu_2 \cdots \mu_k} \right)} & (j = 0) \\ \dfrac{\lambda_0 \lambda_1 \cdots \lambda_{j-1}}{\mu_1 \mu_2 \cdots \mu_j} P_0 & (j = 1, 2, \cdots) \end{cases} \tag{5}$$

It is equivalent to

$$\lambda_j P_j = \mu_{j+1} P_{j+1} \qquad (j = 0, 1, \cdots) \tag{6}$$

Then

$$\sum_{j=0}^{\infty} P_j = 1 \tag{7}$$

λ_j and $\mu_j (j = 0, 1, \cdots)$ are the increasing and decreasing rate of the queue length, respectively. During the time period $(t, t + \Delta t)$, the probability of transferring the queue length from one state to another is $\dfrac{\lambda}{\lambda + j\mu}$. Therefore, in the time of $(t, t + \Delta t)$, the probability of transfer is:

$$[(\lambda + j\mu)\Delta t + o(\Delta t)]\frac{\lambda}{\lambda + j\mu} + o(\Delta t) = \lambda\Delta t + o(\Delta t) \tag{8}$$

where $(\Delta t \to 0)$.

On the left side of formula (8), $o(\Delta t)$ is the transfer increment when the serving process is completed, then

$$P\{L(t+\Delta t) = j+1 \mid L(t) = j\} = \lambda\Delta t + o(\Delta t)\ (\Delta t \to 0) \tag{9}$$

Similarly,

$$P\{L(t+\Delta t) = j\text{-}1 \mid L(t) = j\} = j\mu\Delta t + o(\Delta t)\ (\Delta t \to 0) \tag{10}$$

The probability of no transfer is $e^{-(\lambda + j\mu)\Delta t}$ in the time of $(t, t+\Delta t)$ while the probability of one or more transfer is $o(\Delta t)(\Delta t \to 0)$, then

$$P\{L(t+\Delta t) = k \mid L(t) = j\} = o(\Delta t) \tag{11}$$

where $\Delta t \to 0, |k - j| \geq 2$.

The queue length at the time of t should follow the constraints of increasing rate $\lambda_j = \lambda (j = 0,1,\cdots)$ and decreasing rate $\mu_j = j\mu (j = 0,1,\cdots)$, and

$$P_j = \frac{(\lambda / \mu)^j}{j!} e^{-\lambda/\mu} \qquad (j = 0,1,\cdots) \tag{12}$$

We can calculate the instantaneous average queue length a from the equilibrium probability:

$$a = \sum_{j=1}^{s-1} jP_j + s\sum_{j=s}^{\infty} P_j \tag{13}$$

where s notes the number of the packets in the queue. Since request packets arrive randomly, the congestion state will not change before blocking packets are eliminated, so

$$\lambda_j = \begin{cases} \lambda & (j = 0,1,\cdots,s-1) \\ 0 & (j = s) \end{cases} \tag{14}$$

As the packet delivery ratio and average packet length are exponential distribution, when there are j data packets in accordance with the processing speed μ_j, μ_j is calculated as

$$\mu_j = j\mu \qquad (j = 1,2,\cdots,s) \tag{15}$$

From (14), (15) and (5), we can get the probability of average length exceeding the threshold:

$$P_j = \frac{\frac{(\lambda / \mu)^j}{j!}}{\sum_{k=0}^{s} \frac{(\lambda / \mu)^k}{k!}} \qquad (j = 0,1,\cdots,s) \tag{16}$$

when $Q\_avg > Q\_max$, we add a certain drop probability P_drop.

$$P\_drop = P_j(\tau_1 + \tau_2) \tag{17}$$

and

$$\tau_1 = Q\_avg / Q\_lim \tag{18}$$

$$\tau_2 = Q\_now / Q\_lim \tag{19}$$

The proposed algorithm is presented in Algorithm 1.

```
1:  Initialize Q_avg, Q_min, Q_max
2:  if the node is destination then
3:    Update route table and return RREP
4:  else
5:    if Q_avg < Q_min then
6:      if the node has fresher route to destination
7:          Update route table and return RREP
8:      else
9:        if Q_avg < Q_min then
10:         Update route table and forward RREQ packets
11:       else
12:         Calculate P_delay
13:         Forward RREQ packets with P_delay
14:   else
15:     Calculate P_drop
16:  Free RREQ packets with P_drop
```

Algorithm 1: DRED Algorithm

In the experiment we found that even the randomly dropping method cannot control RREQ queue length effectively in high traffic. The length of RREQ queue will be large for a long time. Traditional expanding ring search has three fixed parameters: initial ring TTL_START, expanding ring $TTL\_INCREMENT$, and the final ring $TTL\_THRESHOLD$. When it is the initialization phase, the initial ring is used to search destination node. And if there is no available route to destination node, the ring value which is the sum of TTL_START and $TTL\_INCREMENT$ will be broadcasted to network until it has reached $TTL\_THRESHOLD$. If there is no available route under the condition of ring value $TTL\_THRESHOLD$, the source node will give up route discovery to the destination node, and consider the destination is unreachable. This method is to control the spreading rate of RREQ packets in the network. Therefore, the method decreases the congestion and overhead in networks. But, it cannot adapt to the congestion condition dynamically due to the fixed value. We set $TTL\_THRESHOLD$ as a variable value according to the fitting curve.

$$TTL_{\_THRESHOLD} = TTL_{\_THRESHOLD} * \left\lfloor \alpha e^{-x/\beta} + f(\lambda) \right\rfloor \tag{20}$$

where x refers to the queue length. So when RREQ queue length is long, we'll inhibit the increase of expanding ring to reduce the spread rate of RREQ message in the network.

The second problem is about the Hello packet interval. In literature [12], Hello packet interval is calculated based on the active degree of the neighbor nodes. The standard to judge the active degree of neighbor nodes is the number of packets transported by neighbor nodes per unit time. In this paper, Hello packet interval is changed with the average queue length. The RREQ queue length will be longer as the average queue length becomes larger. In the case of average queue length becoming larger, the RREQ queue length will be longer, and the

network congestion is more serious. So we should increase the Hello packet interval to make the link more stalwart. When the average queue length is small, network is in good condition with less congestion. At this time, we should reduce the Hello packet interval to decrease unnecessary link maintenance.

The Hello packet interval is determined by:

$$HELLO_{-INTERVAL} = HELLO_{-INTERVAL} * \lfloor \alpha e^{-x/\beta} + f(\lambda) \rfloor \quad (21)$$

Here x also refers to the average queue length, and the Hello packet interval changes with the average queue length. Hello packet lifetime is defined as follows:

$$lifetime = MinHelloInterval + ((MaxHelloInterval - MinHelloInterval) * Random::uniform()) \quad (22)$$

where $MaxHelloInterval$ and $MinHelloInterval$ are multiples of Hello packet interval. Hello packet lifetime follows a uniform distribution which is changed with the Hello packet interval.

4. Numerical Simulations

To evaluate of the network performance, we modified AODV and AOMDV with the proposed mechanism using NS2.35. The mechanism can also be applied to other reactive routing protocols which make route discovery with flooding request and maintain the path link integrity with Hello packets. Simulation settings are shown in Table 2.

Table 2. Simulation Settings

Parameter	Value	Parameter	Value
Propagation model	TwoRayGround	Number of nodes	50
Sending rate	25 per/s	PHY/MAC	802.11(no RTS)
Topology size	1000×300*m*	Pause time	0,50,100,200,300s
Packet length	512 Byte	Mobility model	Random way point
Flows	CBR	Data rate	11Mbps

(1) Node movement model.

The number of nodes is 50. Topology range is a rectangular. The node mobility pause time is 0s, 50s, 100s, 200s and 300s, respectively. And the wireless propagation model is TwoRayGround model. Simulation time is 300s.

(2) Traffic model

Communication traffic model is the stable bit stream transmission model, the number of link is 20. The position of the source node and destination node and the start moving time of nodes are random. The sending rate is 25 packet/s and the packet length is 512 Byte.

(3) Performance evaluation

We select three parameters to evaluate the network performance, i.e., packet delivery ratio, average end-to-end delay and route discovery frequency.

With the communication flow model mentioned above, we set the packet delivery ratio of AODV and AOMDV be 20% according to the curve depicted in Figure 2. And the packet

delivery ratio ranges from 15%~40% according to the confidence interval. We calculate the packet sending rate by formula (3) and obtain the corresponding average queue length by Table 2. Their values are respectively 12~38 per/s and 13~48. Then we set them as thresholds Q_min and Q_max . Finally, we compute the average of them running 20 simulation experiments in different pause time. The results are compared to AODV, AOMDV and literature [3], as shown in Figures 3, 4 and 5.

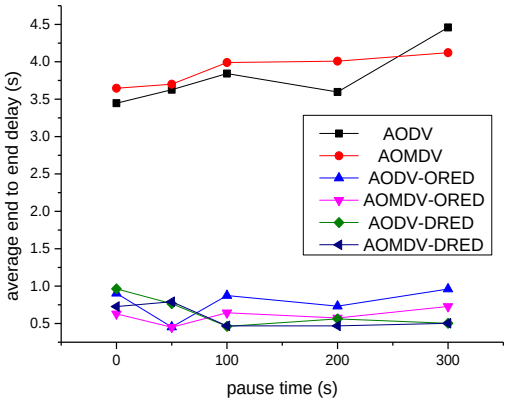

Figure 3. Average end-to-end Delay

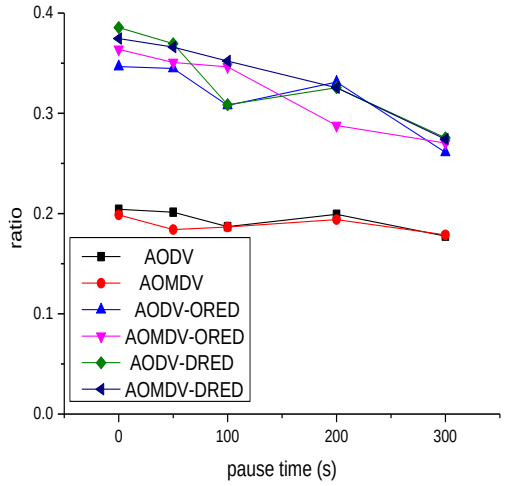

Figure 4. Packet Delivery Ratio

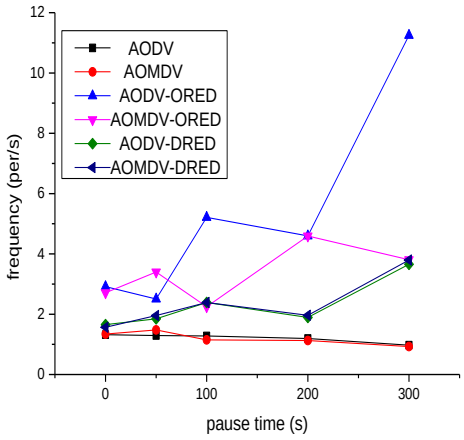

Figure 5. Route Discovery Frequency

Figure 3 shows that the average end-to-end delay is reduced both in the literature [3] and the mechanism proposed in this paper. However, our mechanism has smaller average end-to-end delay when the pause time increases up to 100s and above. This is because when the pause time is small, the nodes move quickly, leading to the severe change of network topology and the increase of broken links. Therefore the RREQ packet is forwarded more frequently from the source node to destination. This will cause severe congestion of RREQ packets. To solve this problem, literature [3] drops all RREQ packets when the length of queue is longer than the threshold values. Our mechanism drops the arriving RREQ packets with a certain probability while increasing a little delay when the nodes move speedily.

Figure 4 shows that our mechanism is very effective in improving the performance of packet delivery ratio. We estimate the packet delivery ratio based on the fitting function curve, and then the corresponding average queue length is set to the threshold. The packet delivery ratio obtained by the experiment results is in the range of our forecast. What is more, it is higher than that in literature [3]. This will provide us an approach to predict the network parameters according to the tolerance values of the packet delivery ratio, and then it makes the whole network process under control.

Figure 5 shows the route discovery frequency. The route discovery frequency will increase slightly owing to the threshold strategy is used to forwarding. The route discovery frequency increases dramatically with the pause time in literature [3], while it keeps steady with the proposed mechanism. It is slightly larger than the original protocols.

In terms of controlling the Hello packet interval, we refer to the parameter settings in [12], as shown in Table 3.

Table 3. Parameter Settings

Parameter	Value	Parameter	Value
Propagation model	Rayleigh fading	Mobility speed	Max 5 m/s, 0 sec pause time
Transmission power	16dBm	Mobility model	Random way point
Topology size	670×670*m*	Flows	ftp Pareto
Data rate	11Mbps	Number of senders	10
Number of nodes	20	PHY/MAC	802.11(no RTS)

We respectively select two parameters Hello packet overhead in different node numbers and the aggregated throughput in different maximum speed to compare with literature [12]. And we calculate the average of them for 100 repeated experiments. The results are shown in Figures 6 and 7.

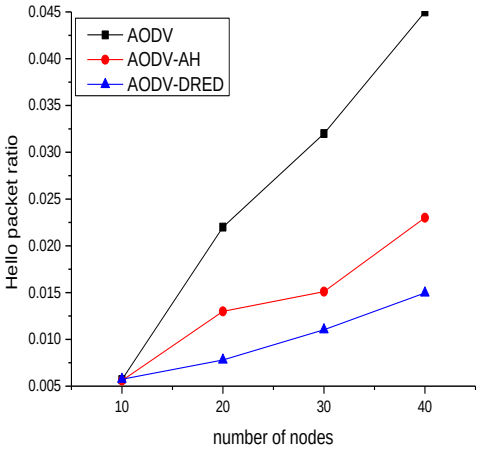

Figure 6. Hello Packet Overhead

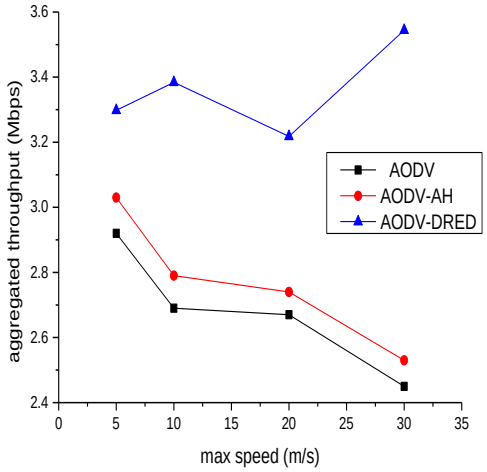

Figure 7. Throughput for Various Max Speed

Figure 6 shows the Hello packet overhead for various node numbers based on the proposed approach compared to that of literature [12]. The Hello packet overhead given by literature [12] is about 50% of the original protocol, and it is given only about 30% by the mechanism proposed in our work. Thus it reduces the Hello packet overhead effectively.

Figure 7 shows the aggregated throughput when the max speed varies. From the result we find that the aggregated throughput is much larger based on our mechanism than that of literature [12]. This indicates that the proposed method of deciding the period of sending Hello packets can more easily adapt to the dynamic network topology and can maintain a

larger network throughput when the nodes move faster which may lead to severe network congestion.

The above simulation results and analysis show that DRED greatly improves the performance in terms of the packet delivery ratio and the average end-to-end delay when the route discovery frequency is increased slightly. The performance of the average end-to-end delay, the packet delivery ratio and the routing discovery frequency are better than those in literature [3], and the Hello packet overhead and the aggregated throughput have been improved considerably compared to literature [12]. Therefore, our system has better performance for reactive routing protocols.

5. Conclusions

In this paper, we mainly solved the problem of REEQ packets congestion in reactive MANET routing protocols when the network traffic is high. First of all, we fit the function curve between the packet delivery ratio and the packet sending rate, and then we analyzed the relation between packet sending rate and average queue length of RREQ queue. According to the average queue length of RREQ queue and the fitting curve, we set the RREQ threshold values and the probability to drop redundancy packets randomly. Finally, we verified the improved mechanism on AODV and AOMDV based on the above theory. The simulation results indicate that the mechanism proposed in this paper avoids obvious increase of routing discovery frequency, achieves smaller average end-to-end delay and Hello packet overhead, and provides better packet delivery ratio. Furthermore, the packet delivery ratio in network is controllable, which provides reference to design the network parameters theoretically according to the tolerance values of an index in MANET networks.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (no. 61103248). The authors are grateful for the anonymous reviewers who made constructive comments and improvements.

References

[1] G. Tomar, T. Sharma and D. Bhattacharyya, "Performance comparison of AODV, DSR and DSDV under various network conditions: A survey", International Conference on Ubiquitous Computing and Multimedia Applications (ICUCMA), Daejeon, Korea, **(2011)** April, pp.3-7.
[2] V. Firoiu and M. Borden, "A study of active queue management for congestion control", IEEE INFOCOM, Tel Aviv, **(2000)** March, pp. 1435-1444.
[3] H. Yousefi, A. Habibi and X. Li, "A statistical study of loss-delay tradeoff for RED queues", IEEE/ACM Trans. Communication, vol. 60, no. 7, **(2012)** July, pp. 1966-1974.
[4] W. Chen, G. Min and H. Zhang, "Statistical adapting RED in dynamic networks", IEEE GLOBECOM, Anaheim, CA, **(2012)** December, pp. 2560-2565.
[5] Z. Li and Z. Jin, "A Wireless Ad Hoc Network Congestion Control Algorithm based on Game Theory", International Conference on Future Computer Sciences and Application (ICFCSA), Hong Kong, China, **(2011)** June, pp. 137 - 141.
[6] F. Klemm, Z. Ye, V. Krishnamurthy and K. Tripathi, "Improving TCP performance in ad hoc networks using signal strength based link management", Ad Hoc Networks, vol. 2, no. 3, **(2005)** March, pp. 175–191.
[7] C. Chen, C. Wong and Y. Kuo, "Signal strength based routing for power saving in mobile ad hoc networks", Journal of Systems and Software, vol. 8, no. 83, **(2010)** August, pp. 1373–1386.
[8] C. Biradar and S. Manvi, "Neighbor supported reliable multipath multicast routing in MANETs", Journal of Network and Computer Applications, vol. 3, no. 35, **(2012)** May, pp. 1074–1085.
[9] G. Park, J. Choi and J. Song, "The residual battery capacity and signal strength based on power-aware routing protocol in mobile ad-hoc network", International Conference on Computer Engineering and Applications (ICCEA), Stevens Point, WI, **(2007)** February, pp. 319-324.

[10] M. Sunita and N. Usturge, "Study of congestion control using AODV and signal strength by avoiding link failure in MANET", International Conference on Communication, Information & Computing Technology (ICCICT), Mumbai, India, **(2012)** October, pp. 1-5.
[11] W. Liu, M. Wu, J. Peng and G. Wang, "The improved ECN congestion control in Ad Hoc network and simulation test", International Conference on Electronic Commerce and Business Intelligence (ICECBI), Beijing, China, **(2009)** June, pp. 143 - 146.
[12] S. Han and D. Lee, "An adaptive Hello messaging scheme for neighbor discovery in on-demand MANET routing protocols", IEEE Communications Letters, vol. 17, no. 5, **(2013)** May, pp. 1040 - 1043.
[13] C. Gomez, A. Cuevas and J. Paradells, "AHR: a two-state adaptive mechanism for link connectivity maintenance in AODV", International Symposium on Mobile Ad Hoc Networking & Computing (ISMANC), New York, NY, **(2006)** May, pp. 98-100.
[14] F. Ingelrest, N. Mitton and D. Simplot, "A turnover based adaptive HELLO protocol for mobile Ad Hoc and sensor networks", International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS), Istanbul, Turkey, **(2007)** October, pp. 9-14.
[15] P. Razafindralambo and N. Mitton, "Analysis of the impact of hello protocol parameters over a wireless network self-organization", International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (IWMASWM), Crete Island, Greece, **(2007)** May, pp. 46-53.
[16] S. Suman and Balkrishan, "Enhanced AODV for wireless networks", Advance Computing Conference (IACC), Patiala, India, **(2010)** February, pp. 246-249.
[17] M. Dadhania and K. Vinay, "Modified RED algorithm to improve the performance of web traffic", International Conference on Advanced Computing & Communication Technologies (ICACCT), Rohtak, India, **(2013)** April, pp. 187-194.
[18] X. Jiang, J. Yang and GX. Jin, "RED-FT: A scalable Random Early Detection scheme with flow trust against DoS attacks", IEEE Communication Letter, vol. 17, no. 5, **(2013)** May, pp. 1032 - 1035.
[19] J. Zhao, S. Guo and K. Zheng, "A congestion control scheme based on improved RED algorithm in internet topology simulation", International Conference on Computer Science & Service System (ICCS), Nanjing, China, **(2012)** August, pp. 279-283.
[20] T. Wilson, G. Chen and T.L, "RED-f routing protocol for complex networks", International Symposium on Circuits and Systems (ISCS), Seoul, Korea (South), **(2012)** May, pp. 1644-1647.
[21] I. Jarvinen, Y. Ding and A. Nyrhinen, "Harsh RED: improving RED for limited aggregate traffic", International Conference on Advanced Information Networking and Applications (AINA), Fukuoka, Japan, **(2012)** March, pp. 832-840.
[22] F. Nakamura and T. Nakashima, "Performance of a stable unit active queue management", International Conference on Intelligent Networking and Collaborative Systems (INCS), Fukuoka, Japan, **(2011)** November, pp. 508-513.
[23] B. Abbasov, "Using linear interpolation method for packet drop probability function of RED algorithm", International Conference on Application of Information and Communication Technologies (AICT), Baku, **(2011)** October, pp. 1-4.

Authors

Wentao Wang, he received M.S. degrees in Computer Application Technology from South Central University for Nationalities, Wuhan, China, in 2001, and received the Ph.D. degree in Control science and Engineering from Huazhong University of Science and Technology in 2010. Also, from 2002 to 2003, he worked as a visiting scholar at department of Computer Engineering, Chonbuk National University, South Korea. His current research interests include mobile and sensor networks, image processing, and computational intelligence.

Hao Wang, he received bachelor degree in electronic information science and technology from the Henan Polytechnic University in 2011 and now studying for his master Degree in computer science and application in South-Central University for Nationalities. His research interesting includes computer networks, multimedia.

Wan Tang, she received the B.S. and M.S. degrees in Computer Application Technology from South Central University for Nationalities, Wuhan, China, in 1995 and 2001, respectively, and received the Ph.D. degree in Communication and Information System from Wuhan University, China in 2009. She is currently an associate professor in the School of Computer Science of South-Central University for Nationalities. Also, from 2001 to 2002, she worked as a visiting scholar at department of Computer Engineering, Chonbuk National University, South Korea. Furthermore, from 2012 to 2013, she worked as a visiting scholar at department of Computer Science and Engineering, SUNY at Buffalo, USA. Her research interests include protocols for optical/wireless communication networks, network security, and computational intelligence.

Feng Guo, he received bachelor degree in computer science and application from South-Central University for Nationalities in 2012 and now studying for his master Degree in computer science and application in South-Central University for Nationalities. His research interesting includes computer networks.

