http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (2): 255-261, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.255.261
© 2012 Asian Network for Scientific Information

Performance Assessment of AGRED, RED and GRED Congestion Control Algorithms

[1]Mahmoud Baklizi, [2]Hussein Abdel-jaber [1]Sureswaran Ramadass,
[1]Nibras Abdullah and [1]Mohammed Anbar
[1]National Advanced IPv6 Centre (Nav6), Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
[2]The World Islamic Sciences and Education WISE University, Amman, 11947, P.O. Box 1101, Jordan

Abstract: With the recent rapid development in computer networks, congestion has become a critical issue. As a network term, congestion usually occurs when the incoming packets exceed the available network resources. This study proposed an Adaptive Gentle Random Early Detection (AGRED) algorithm based on Gentle Random Early Detection (GRED) algorithm in order to detect congestion at router buffer at a preliminary stage and thus enhance the parameter setting of the max threshold and the D_{max}. The AGRED algorithm is simulated and compared with the original GRED and Random Early Detection (RED). The simulation results for the proposed AGRED, RED and GRED algorithms are carried out by varying the variable of packet arrival probability. During congestion, the simulation results reveal that AGRED offers marginally better performance results than RED and GRED, with regard to mean queue length, average queuing delay and packet loss probability due to overflow. Therefore, the AGRED is an effective method in controlling congestion at router buffers of networks. Whereby, improve networks performance.

Key words: Queuing network, discrete-time queue, active queue management, performance evaluation, packet loss

INTRODUCTION

Network congestion causes increase packet loss rate, increase queuing delay for the packets loss and reduce the throughput rate. Whereby, degrades the network performance (Abdelouahab *et al*., 2006; Thiruchelvi and Raja, 2008; Jasem *et al*., 2011; Welzl, 2005). Congestion occurs at router buffers of networks when the amount/number/size of incoming packets exceeds the available network resources (Tanenbaum, 2002; Babainejad *et al*., 2010; Jasem *et al*., 2010). This can play a key role in the deterioration of computer networks (Sasipraba and Srivatsa, 2006; Chen *et al*., 2010) includes the following:

- Increasing the packet dropping probability (D_p)
- Growing/Maximizing the packet loss probability due to overflow (PL) result
- Increasing the mean queue length (mql) for packets
- Increasing the mean waiting time (D) for packets Degrading the amount of packet passing through the router buffer successfully (throughput (T))

Active Queue Management (AQM) algorithms have been proposed to improve the network performance (Feng *et al*., 2002; Ayanzadeh *et al*., 2009; Ababneh *et al*., 2010). Examples of AQM algorithms are RED (Floyd and Jacobson, 1993), Gentle RED (Floyd, 2000), Adaptive RED (Floyd *et al*., 2001), Random Early Marking (REM) (Athuraliya *et al*., 2001), Dynamic Random Early Drop (DRED) (Aweya *et al*., 2001) and some discrete-time queue analytical models (Abdel-Jaber *et al*., 2007, 2008; Ababneh *et al*., 2010) which are built based on some AQM methods. For instance, the analytical models of DRED, GRED and BLUE were built by analyzing a single queue node using a discrete time queue approach (Daduna, 2001; Shakiba *et al*., 2008).

One of the most known AQM methods is RED whose performance may worsen in many situations. For example, at a definite time, the arrival rate may well increase and as a corollary the RED router buffer builds up and overflows. The congestion measure of RED (aql) could be smaller than the minimum threshold position at the router buffer (min. threshold). Thus, no packets will be dropped even though the RED router buffer is overflowing. Another obstacle of RED is its dependency on the input setting (min. threshold, max. threshold, queue weight (qw), D_{max}) (Floyd and Jacobson, 1993). These parameters must be set to certain values in order to derive a satisfactory performance (Floyd *et al*., 2001; Chen *et al*., 2010). One

Corresponding Author: Mahmoud Baklizi, National Advanced IPv6 centre (Nav6), 6th Floor, School of Computer and Mathematical Sciences Building, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia

potential solution to prevent RED's router buffer from building up rapidly is to utilize the instantaneous queue length as a congestion detector rather than the aql. This gives the former the opportunity to go beyond the min. threshold position and to drop packets probabilistically/probably before the router buffer overflows. GRED, BLUE and DRED algorithms use the instantaneous queue length as a congestion detector. In particular, the GRED algorithm was primarily proposed to deal with the aforementioned limitations of RED. Indeed, GRED can improve the process of setting the max threshold and the D_{max} parameters and is able to stabilise the aql at a position named T_{aql}. The T_{aql} position is half way from the min threshold and max threshold positions which prevents the router buffer from filling up and becoming larger than the max threshold position and as a result fewer packets are dropped.

In this study, a new algorithm called AGRED based on GRED aimed to enhance the performance of GRED with reference to mql, D and P_L performance measures chiefly during congestion. The D_{init} value of the AGRED algorithm varies from Dmax to 0.5 and the aql varies from max threshold to double max threshold. On the other hand, the Dinit value of GRED varies from D_{max} to 1.0 and the aql value varies from max threshold to double max threshold. This enables the proposed algorithm to provide further enhancements in setting the input parameters, e.g., max threshold and D_{max}. In addition, we compare the original GRED and RED with our proposed AGRED model. During congestion, the simulation results reveal that the AGRED drops fewer packets than RED and GRED and it marginally offers better performance results than RED and GRED.

RANDOM EARLY DETECTION (RED)

RED algorithm has been proposed by Floyd and Jacobson (1993) for one of the most known AQM. One goal of using the RED algorithm is to detect instantaneous congestion. Another goal is to avoid congestion by controlling the average queue size for the router in a region of low delay and high throughput (Al-Nabhan *et al*., 2006; James Abu and Gordon, 2011). The aql is compared to two thresholds. If the aql value is smaller than minimum threshold, no packets will be dropped. On the other hand, if the aql value is larger than maximum threshold, a heavy congestion occurs and the arriving packet is marked. Finally, when the aql is between minimum and maximum thresholds, the congestion occurs and the router drops the packet with the probability of D_p (Floyd and Jacobson, 1993).

GENTLE RANDOM EARLY DETECTION (GRED) METHOD

GRED was proposed by Floyd (2000) to deal with some of RED's issues (Floyd, 2000; Aweya *et al*., 2001; Floyd *et al*., 2001). The pseudocode of GRED is shown in Figure 1 and a description (definition) of the parameters used in the pseudocode is presented in Table 1.

Figure 1 shows that every time a packet arrives at the router buffer, the aql value is calculated as in RED (Floyd and Jacobson, 1993), as given in the following equation:

$$aql = aql x (1 - qw) + q\_instantaneous^{x}\ qw \tag{1}$$

If the aql value is smaller than the min threshold position, no packets will be dropped and the D_p value is set to zero. On the other hand, if the aql value is larger than or equal to double max threshold position, a severe congestion occurs and afterward every arriving packet will be dropped with D_p =1. Finally, when the aql value is between min threshold and double max threshold positions, a congestion occurs and the router buffer drops the arriving packets probabilistically ($0<D_p<1$). The result of D_{init} varies between D_{max} and 1 as long as the aql value varies between max threshold and double max threshold positions (Floyd, 2000). This variance produces further harmony to the parameter settings of max

Table 1: Description of the parameters used

Definitions	Description
Current time	The current time
Idle time	The beginning waiting time at the router buffer
n	The number of packets transmitted to the router buffer through an idle interval time
C	A connter that represents the number of packets arrived at the router buffer and has not dropped since the last packet was dropped
D_p	The packet dropping probability
D_{init}	The initial packet dropping probability
q_instantaneous	The instantaneous queue length
qw	The queue weight
D_{max}	The maximum value of D_{init}
q (time)	The linear function for the time
T_{aql}	Target level for the aql
Double max threshold	It is set to 2 x max threshold

```
Step 1: Initialization stage
C = -1;
aql = 0.0;

Step 2: for every arriving packet at a GRED router buffer:
Calculate the aql for the arriving packet at the router buffer.
Examine the queue status at the router buffer, e.g., empty or not. if (the queue at the router buffer = = empty)
{Compute n, where n = q(current _ time - idle _ time);
aql = aql x (1-qw) n; } else
aql = aql x (1-qw)+qw x q_instantaneous;

Step 3: Determining the congestion status at the router buffer:
if (aql<min threshold) {
D = 0.0; // No packets have dropped
Set C = — 1;
}
else if (min threshold<aql and andaql<max threshold)
{
C = C+1;
calculate the $D_p$ value for the arriving packet as follows:

    $D_{init} = \frac{D_{max} \times (aql - \text{min threshold})}{\text{max threshold} - \text{min threshold}}$

    $D_P = \frac{D_{init}}{(1 - C \times D_{init})}$

drop arriving packet probabilistically in terms of its $D_P$ value;
Set C = 0 ;
}
else if (max threshold = aql and andaql<double max threshold)
{
C = C+1;
calculate the dropping probability ($D_P$) for the arrival packet as follows:

    $D_{init} = D_{max} + \frac{(1 - D_{max}) \times (aql - \text{max threshold})}{\text{max threshold}}$

    $D_P = \frac{D_{init}}{(1 - C \times D_{init})}$

Mark/drop arriving packet probabilistically in terms of its ($D_P$) value;
Set C = 0 ;
}
else // if (aql = double max threshold)
{
 Mark/drop every arriving packet with $D_P$ = 1;
Set C = 0 ;
}

Step 4: When the GRED router buffer becomes empty
 Set idle time = current time;
```

Fig. 1: The pseudocode of GRED algorithm

threshold and D_{max}. The parameters of GRED, i.e. ,q_W and min threshold have been set as in RED.

THE PROPOSED AGRED

An AGRED algorithm is proposed to improve the performance of GRED during congestion situations, i.e., deriving better quality results with reference to mql, D and PL performance measures. Also, the proposed algorithm aims to enhance the parameter settings, e.g., max threshold and D_{max}, of GRED. The AGRED drops the arriving packets incipiently during the congestion. The congestion measure of the AGRED is aql and its parameters are similar to those of GRED. The calculation

```
Step1: Initialization stage
C = -1;
aql = 0.0;

Step 2: for every arriving packet at an AGRED router buffer:
Calculate the aql for the arriving packet at an AGRED the router buffer.
Examine the queue status at the router buffer whether empty or not.
 if (the queue at the AGRED router buffer = = empty)
{
Compute n, where n = q (current _ time - idle _ time);
aql = aql x $(1\text{-qw})^n$;
 }
 else
aql = aql x (1-qw)+qw x q_instantaneous;

Step 3: Determining the congestion status at the AGRED router buffer:
if (aql<min threshold) {
$D_P$ = 0.0; // No packets have dropped
Set C = — 1;
 }
else if (min threshold = aql and and aql<max threshold)
{
C = C+1;
calculate the $D_p$ value for the arriving packet as follows:

    $D_{init} = \frac{D_{max} \times (aql - \text{min threshold})}{\text{max threshold} \quad \text{min threshold}}$

    $D_P = \frac{D_{init}}{(1 - C \times D_{init})}$

Mark/drop arriving packet probabilistically in terms of its $D_p$ value;
Set C = 0 ;
}
else if (max threshold = aql and andaql<ouble max threshold)
{C = C+1;
calculate the dropping/probability ($D_p$) for the arrival packet as follows:

    $D_{init} = D_{max} + \frac{\frac{(1 - D_{max})}{2} \times (aql - \text{max threshold})}{\text{max threshold}}$

    $D_P = \frac{D_{init}}{(1 - C \times D_{init})}$

Mark/drop arriving packet probabilistically in terms of its ($D_p$) value;
Set C = 0 ;
}
else // if (aql = double max threshold)
{
 Mark/drop every arriving packet with $D_p$ = 1;
Set C = 0 ;
}

Step 4:When the GRED router buffer becomes empty
Set idle time = current time;
```

Fig. 2: The pseudocode of the proposed AGRED method explained in detail

of the aql in the proposed algorithm is also similar to that of GRED. Therefore, the AGRED decides whether to drop every arriving packet as in the GRED algorithm (Fig. 1, 2). The pseudocode of AGRED is presented in Fig. 2.

The main difference between GRED and the AGRED is in calculating the D_{init} value. The way of computing D_{init} in GRED is shown in Fig. 1. the AGRED computes the D_{init} according to the following equation:

$$D_{init} = D_{max} + \frac{(1 - D_{max})}{2} \times \frac{(aql - \text{max threshold})}{\text{max threshold}} \quad (2)$$

Put simply, when aql value is between max threshold and double max threshold positions, the calculated D_{init} value of GRED varies from D_{max} value to 1.0 as the aql value varies from max threshold to double max threshold position. However, in the AGRED, the D_{init} value increases from D_{max} value to 0.5 as long as the aql value increases from max threshold to double max threshold position. This further improves the parameter settings of max threshold and D_{max} than those of GRED.

The proposed AGRED makes its router buffer overflow at times lower than those of GRED. Lastly, the running time of the AGRED is the amount of time of all events. i.e., packet arrivals and departures.

SIMULATION INFORMATION OF RED, GRED AND AGRED

It has been assumed that a denotes the packet arrival probability at the router buffer in a fixed time unit called slot (Daduna, 2001) and β denotes the probability of packet departure from the router buffer in a slot. The packet arrivals can be modeled using a Bernoulli process and packet departures can be modeled using a geometrical distribution. The geometrically distributed means $1/\alpha$ and $1/\beta$ are used for the packet inter-arrival times and service times, respectively.

GRED, RED and the proposed method are simulated based on discrete-time queue (Abdel-Jaber *et al*., 2008) that uses a slot as a time unit. In each slot, packet arrival and/or departure may exist. The compared algorithms are simulated by applying them in a network consisting of a single router buffer node (Fig. 3). It should be noted that Packet arrival and departure are implemented in a single mode. The scheduling mode is First Come First Served (FCFS). The GRED, RED and the AGRED simulations are implemented in Java, on i7 processor machine with 1.66 GHz and on 4 GB RAM.

PERFORMANCE EVALUATION RESULTS

This section compares the original GRED algorithm, RED and the AGRED with reference to different performance measures, e.g. (mql, T, D, P_L and D_p) in order to determine which algorithm offers better performance. The parameters of the compared methods are similar and have been set as follows: α and β were set to [0.18-0.93] and 0.5, respectively intending to create noncongestion ($\alpha<\beta$) and congestion ($\alpha>\beta$) situations. The maximum number of queue node places (K) is limited and is equal to 20 and thus this number should generate accurate performance measure results at small queue sizes. min threshold, max threshold, D_{max} and qw have been set to 3, 9, 0.1 and 0.002, respectively as in RED (Floyd and Jacobson, 1993). double max threshold is set to 18 as in GRED (Floyd, 2000). In the simulation, a large number of slots have been used (2000000) in order to generate a warming-up period which ends when the system reaches a steady state.

The performance measure results are evaluated by the changeable values of α. Hence, the assessment as to which method gives better performance is only determined based on the values of α. After the system reaches the steady state, the evaluation of performance measures can be achieved. For each α, the simulations are run ten times in which in each run, the seed value for the random number generator is changed aiming to delete a bigotry in the performance measure results. Also, the performance measure result for each α represents the mean of ten run times for that value. All performance measure results versus α values are shown in Table 2 and Fig. 4-5. Table 2 displays mql, T and D results of the RED, GRED and the AGRED versus α. while the results of P_L and D_p versus α are illustrated in Fig. 4 and 5, respectively.

Table 2 shows that the RED, GRED and the AGRED give similar mql, T, D, P_L and D_p results when no congestion situation (α = 0.18 or 0.33) is occurring since no packets were dropped. In addition, when a light congestion exists both RED and AGRED methods offer

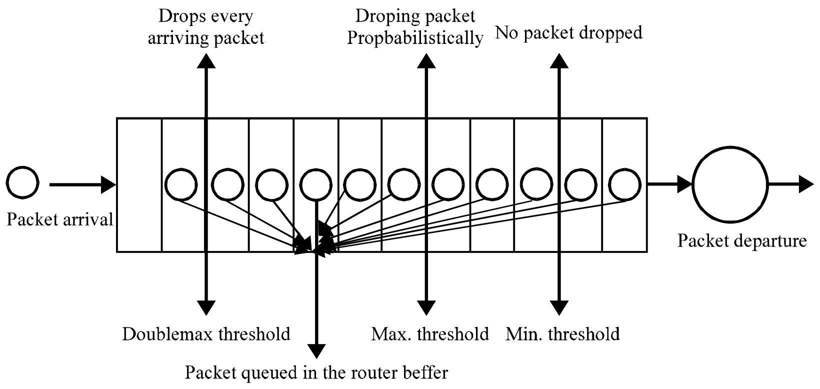
Drops every arriving packet
Droping packet Propbabilistically
No packet dropped
Packet arrival
Packet departure
Doublemax threshold
Max. threshold
Min. threshold
Packet queued in the router beffer

Fig. 3: The single router buffer for RED, GRED and proposed AGRED

Table 2: mql, T and D performance results of RED, GRED and proposed AGRED

	RED			GRED			AGRED		
α	mql	T	D	mql	T	D	mql	T	D
0.18	0.457	0.178704	2.5604	0.457	0.1787	2.5604	0.457	0.17866	2.5601
0.33	1.279	0.3277	3.9030	1.279	0.3277	3.903	1.279	0.3277	3.9038
0.48	6.082	0.4688	12.973	6.1005	0.4689	13.009	6.088	0.4688	12.984
0.63	13.367	0.4964	26.924	13.578	0.497	27.299	13.3645	0.4975	26.861
0.78	15.8307	0.4979	31.792	14.7936	0.49885	29.6551	14.3547	0.49909	28.761
0.93	16.8279	0.4985	33.7548	14.9456	0.499	29.9316	14.4275	0.4995	28.8811

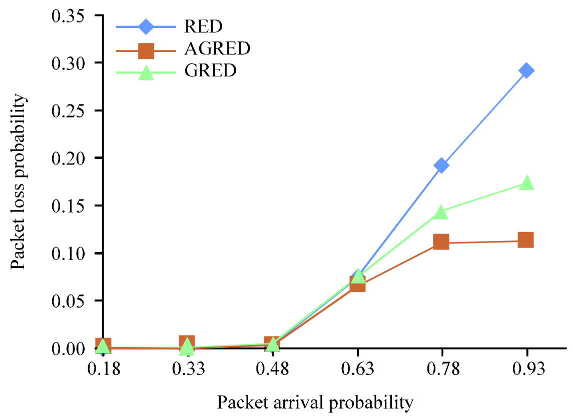

Fig. 4: Packet Loss Probability vs. Packet Arrival Probability

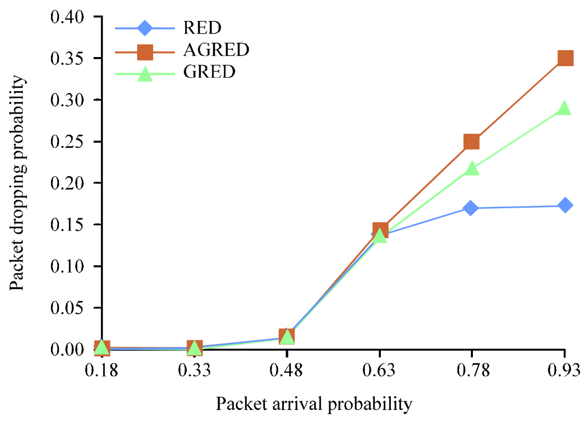

Fig. 5: Packet Dropping Probability vs. Packet Arrival Probability

slightly better mql and D results than those of GRED since few packets were dropped but all methods give similar T results.

Also, if heavy congestion occurs (α>β), the AGRED or GRED offers slightly better mql and D results than RED. This happens due to the fact that the AGRED's router buffer drops more packets than GRED and RED in these circumstances, i.e., α>β (Fig. 5). It can be observed from Table 2 that the compared algorithms generate similar T results regardless of the congestion situation.

Figure 4 shows that the proposed algorithm loses fewer packets due to overflow (P_L) than RED or GRED in congestion situations and this is because it drops larger number of packets (D_p) than GRED and RED (Fig. 5). This gives marginally smaller results of mql and D for AGRED than those of GRED and RED. GRED's router buffer maintains its aql at a value lower than that of the AGRED when a heavy congestion has occurred. Finally, GRED produces smaller P_L results than RED. Yet, at high congestion, RED provides better D_p results than either AGRED or GRED which also drops fewer packets than the AGRED.

CONCLUSIONS AND FUTURE WORK

An AGRED algorithm based on GRED algorithm is presented in this paper to improve the performance of GRED when a congestion situation occurs in the network router buffers. The proposed AGRED algorithm slightly provides better mql, P_L and D results than GRED and RED when a heavy congestion is existed. Furthermore, the AGRED improves the process of parameter settings of max threshold and D_{max}. This is accomplished by varying the D_{init} value from the value of D_{max} to 0.5 as long as the value of aql (its congestion measure) from max threshold value to 2×max threshold value. A comparison between RED, GRED and the AGRED was conducted with the following results:

The methods of RED, GRED and AGRED produced similar mql, T, D, P_L and D_p results with no or light congestion situation. The AGRED somewhat/ rather gave better mql and D results than RED or GRED in the occurrence of high congestion. The compared methods offer similar T results in the existence of a congestion situation. The AGRED yields better PL results than those of GRED or RED when high congestion arises, whereas GRED produces smaller P_L results than RED. Moreover, at high congestion, RED could provide better D_p results than either AGRED or GRED and also GRED drops fewer packets than AGRED.

For the near future, we suggest applying the AGRED method to base stations of cellular networks through a congestion control approach.

REFERENCES

Ababneh, J., H. Abdel-Jaber, F. Thabtah, W. Hadi and E. Badarneh, 2010. Derivation of three queue nodes discrete-time analytical model based on DRED algorithm. Proceedings of the 7th International Conference on Information Technology: New Generations, April 12-14, 2010, Las Vegas, Nevada, USA., pp: 885-890.

Abdel-Jaber, H., M. Woodward, F. Thabtah and M. Al-diabat, 2007. Modelling blue active queue management using discrete-time queue. Proceedings of the World Congress on Engineering, July 2-4, London, pp: 568-573.

Abdel-Jaber, H., M. Woodward, F. Thabtah and A. Abu-Ali, 2008. Performance evaluation for DRED discrete-time queueing network analytical model. J. Network Comput. Appl., 31: 750-770.

Abdelouahab, A., A. Mueen and F. Taibi, 2006. On the support of multimedia traffic over optical burst-switched networks. Inform. Technol. J., 5: 1012-1017.

Al-Nabhan, M., S. Yousef and J. Al-Saraireh, 2006. TCP protocol and red gateway supporting the QoS of multimedia transmission over wireless networks. Inform. Technol. J., 5: 689-697.

Athuraliya, S., S.H. Low, V.H. Li and Q. Yin, 2001. REM: Active queue management. IEEE Network, 15: 48-53.

Aweya, J., M. Ouellette and D.Y. Montuno, 2001. A control theoretic approach to active queue management. Comput. Networks, 36: 203-235.

Ayanzadeh, R., E. Shahamatnia and S. Setayeshi, 2009. Determining optimum queue length in computer networks by using memetic algorithms. J. Applied Sci., 9: 2847-2851.

Babainejad, S., S. Babainejad, N. Bigdeli and K. Afshar, 2010. Setting up a virtual laboratory for evaluation of congestion control algorithms in TCP/IP networks. Trends Applied Sci. Res., 5: 177-187.

Chen, J., C. Hu and Z. Ji, 2010. An improved ared algorithm for congestion control of network transmission. Math. Problems Eng., 2010: 1-14.

Daduna, H., 2001. Queueing Networks with Discrete Time Scale: Explicit Expressions for the Steady State Behavior of Discrete Time Stochastic Networks. Springer, Hamburg, Germany, ISBN-13: 9783540423577, Pages: 138.

Feng, W.C., K.G. Shin, D.D. Kandlur and D. Saha, 2002. The BLUE active queue management algorithms. IEEE/ACM Trans. Networking, 10: 513-528.

Floyd, S. and V. Jacobson, 1993. Random early detection gateways for congestion avoidance. IEEE/ACM Transa. Networking, 1: 397-413.

Floyd, S., 2000. Recommendations on using the gentle variant of RED. http://www.aciri.org/floyd/red/gentle.html.

Floyd, S., R. Gummadi and S. Shenker, 2001. Adaptive RED: An algorithm for increasing the robustness of RED's active queue management. AT&T Center for Internet Research at ICSI. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.78.7450&rep=rep1&type=pdf.

James Abu, A. and S. Gordon, 2011. A framework for dynamically selecting gain in LEDBAT algorithm for high speed networks. Inform. Technol. J., 10: 358-366.

Jasem, H.N., Z.A. Zukarnain, M. Othman and S. Subramaniam, 2010. The delay with new-additive increase multiplicative decrease congestion avoidance and control algorithm. Inform. Technol. J., 4: 1327-1335.

Jasem, H.N., Z.A. Zukarnain, M. Othman and S. Subramamiam, 2011. Efficiency and fairness of new-additive increase multiplicative decrease congestion avoidance and control algorithm. J. Applied Sci., 11: 438-449.

Sasipraba, T. and S.K. Srivatsa, 2006. Network border patrol, a novel congestion avoidance mechanism for improving QOS in wireless networks. Inform. Technol. J., 5: 427-432.

Shakiba, M., M. Teshnehlab, S. Zokaie and M. Zakermoshfegh, 2008. Short-term prediction of traffic rate interval router using hybrid training of dynamic synapse neural network structure. J. Applied Sci., 8: 1534-1540.

Tanenbaum, A.S., 2002. Computer Networks. 4th Edn., Prentice Hall, USA., ISBN 0-13-066102-3.

Thiruchelvi, G. and J. Raja, 2008. A survey on active queue management mechanisms. Int. J. Comput. Sci. Network Secur., 8: 130-145.

Welzl, M., 2005. Network Congestion Control: Managing Internet Traffic. John Wiley and Sons, Chichester, UK., ISBN-13: 9780470025284, Pages: 263.