Asian Journal of Information Technology 14 (6): 221-230, 2015
ISSN: 1682-3915

Light Weight Distributed QoS Algorithm for Wide Area Ad Hoc Networks

Seyed Hossein Hosseini Nazhad Ghazani
Department of Computer Engineering, Ahar Branch, Islamic Azad University, Ahar, Iran

Abstract: Ad hoc networks attract a considerable attention due to their simplicity, low cost and efficiency. Using multimedia applications on these networks will be more widespread in the future. In contrast with best-effort flows, real-time ones seriously need Quality of Service (QoS) support. This study proposes a completely distributed algorithm to provide QoS in ad hoc networks. The proposed algorithm, entitled Light Weight distributed QoS Algorithm (LWQA), using queue data structure and linear algebra, dynamically adjusts Contention Window (CW) related to the flows. Furthermore, LWQA utilizes static or low speed nodes during routing of real-time flows to increase their QoS. In addition to considering flows' priorities, the proposed algorithm is able to distinguish between flows of the same type. We proved the correctness of this algorithm using markov's mathematical model and implemented it in a simulation environment using the Network Simulator Version 2 (NS-2) Software. The simulation results demonstrate that LWQA improves the QoS in ad hoc networks.

Key words: Wide area ad hoc networks, quality of service, back-off time, contention window, delay, real-time flows, distributed algorithms, priority, simulation, node speed

INTRODUCTION

In modern world networking, the computational resources and data sharing are necessary to achieve fast accessibility. Due to the rapid growth and connection capability of electronic equipment, the utilization of ad hoc network is increasing. As ad hoc networks are spreading considerably, they are exploited for data sharing and transmission of multimedia applications as well as normal uses (Preveze and Safak, 2012). The applications require different level of Quality of Service (QoS) provision to them in order to meet user satisfaction. However, the mobility of the communicating nodes, leading to rapidly changing network topology, make QoS support a very complex process. Several distributed algorithms while imposing minimum overload on the network and attempting to perform the tasks with the minimum cost and efficient resource utilization have been designed for QoS support in ad hoc networks. In this category, Mangold *et al.* (2002) have designed algorithms that exploit static parameters such as Contention Window (CW), frame size and Inter-Frame Space (IFS) for different types of flows. The weaknesses of these algorithms are due to lack of attention to the current status of the network, flows and lack of optimal use of available resources in the network.

As opponents of using static parameters, Wang *et al.* (2008), Budyal and Manvi (2013) and Elizarraras *et al.* (2014) dynamically assigned the priority of flows to support QoS in ad hoc networks but neglected the current status of the flows. Therefore, these algorithms are unable to distinguish between flows of the same type. To exemplify this inability; two real-time flows that reach the destination in one second are taken into account where the first flow passes ten nodes whereas the second one passes three. These two flows are assumed to collide in the beginning. In this case, the existing algorithms will treat two flows in the same way as they are of the same type. That is to say, they set a similar back-off time for both flows. However, the back-off time of the first flow must be smaller than that of the second one, due to its longer path. On the other hand, Diaz *et al.* (2014) have designed algorithms to provide QoS in ad hoc networks that are usable in the networks with a few number of nodes where the number of real-time flows does not exceed a specified range. The weakness of them lies in their poor admission control; therefore, QoS of existing flows is degraded by contention to new flow. Along with the admission control utilization, resource reservation is one of the QoS support solutions in ad hoc networks.

Bouhouche and El-Fatmi (2010) have introduced frameworks that use admission control along with resource reservation to provide better services for high priority flows. The resource reservation method has its challenges, i.e., some control signals are utilized for resource reservation. These signals compete with data packets inside the network and as a consequence of

collision between control and data packets, the overall system efficiency decreases. Node mobility is another problem of this method. It is essential that the source node reserve the resources to send high priority data packets. As the nodes are mobile, after reservation, they may move and exit resource reservation area. Therefore, it causes the reserved sources to become useless for a time span which in turn, decreases network efficiency.

Furthermore, Abbas *et al*. (2012), Vijayalakshmi and Ramamoorthy (2013) and Zhang *et al*. (2015) have designed fully-distributed algorithms to QoS support in ad hoc networks. These algorithms do not consider the speed of nodes which generate the paths during routing process. Therefore, high speed nodes have the equivalent probability of incorporation in a communication path as static or low speed nodes. It is evident that the paths, composed of high speed nodes have lower stability and are probable to be broken. Also, a fully-distributed algorithm entitled Faired and QoS Assured Media Access Control (MAC) Protocol for Multi-hop Ad hoc Network (MFQMAC) has been introduced by Seth *et al*. (2013). It assures QoS through service differentiation among different classes of flows and provides fairness among traffic flows of same priority class. The major problems of MFQMAC are as follows: it does not consider the speed of nodes during routing process, the current status of the flows in calculation of back-off time and the current status of the network for admission control.

In the following part, the proposed algorithm is introduced. It overcomes the mentioned problems in a distributive manner using dynamic calculation of both CW and back-offtimes with regard to the flows' statuses as well as the current status of the network and it utilizes static and low speed nodes during routing real-time flows.

MATERIALS AND METHODS

The proposed algorithm aims to provide QoS for real-time flows against best-effort flows in the ad hoc networks. Video transmission over the ad hoc networks is usually interrupted by video packet loss, caused by interference (Ghazali and Harun, 2012). As it is known, much resource is required for transfer real-time flows; consequently, in order to make optimal use of network resources, transmission of real-time flows must be done with greater accuracy with minimal loss and collision. The accuracy of LWQA algorithm was increased utilizing dynamic calculation of CW and involving network and flow status in calculation of back-off times. Also, the assumptive network consists of fast, slow and static nodes. Therefore, during routing process, the speed of nodes is taken into consideration in this way, real-time packets are transmitted via static or low mobility nodes.

Creation of a stable transmission path for real-time flows by Create_Safe_Path() module: When transmitting data, the utilization of slow and static nodes decreases the probability of changes in the selected paths and at the same time, improves the quality of flows; for that reason, it is suggested that during the process of path selection for real time flows, only slow and static nodes respond to an assumed request by forwarding RREQ or RREP packet. In the proposed framework, Create_Safe_Path() module utilizes modified AODV routing protocol. In the modified AODV, slow and static nodes are utilized for sending real-time packets and high speed nodes are utilized for sending non real-time packets. AODV routing protocol consists of two sections; Request/Reply (Perkins and Belding-Royer, 2003; Abdullah *et al*., 2009). As shown in Fig. 1 in request stage, the source node transmits Route Request (RREQ) message for the new flow. RREQ packet includes QoS information such as flow class, needed quality for flow, the minimum operational power or accumulated delay in all previous nodes. Each intermediate node writes information of RREQ packet in its routing table as soon as it receives the message.

As long as the average of node speed for providing services is appropriate for received type of data packet and basically if the flow is locally acceptable in receiver node, it will broadcast the RREQ message again. Inversely, if the intermediate node is not able to fulfill flow requirements, it eliminates RREQ packet. The intermediate nodes announce the potential load by broadcasting Neighbor Reply (NREP) message which is illustrated by dashed line in Fig. 1. The flow information broadcasted by NREP is used as the input of the model. The RREQ packet will reach the destination node if a path with required quality exists.

In the reply stage, destination node transmits Route Reply (RREP) message in the inverse direction toward source node as depicted in Fig. 2. When the RREP

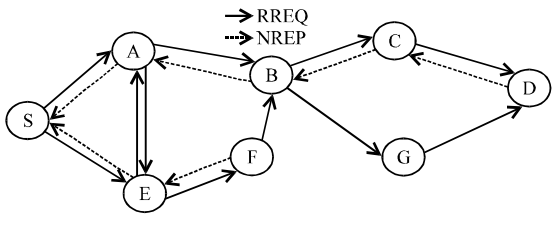

Fig. 1: Request stage

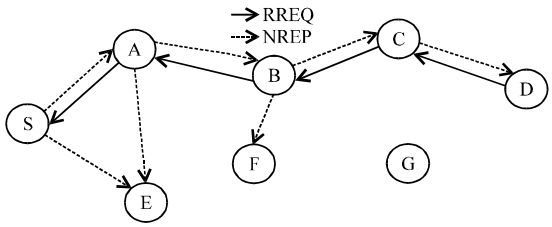

Fig. 2: Reply stage

message is transmitted, intermediate nodes update the load information of the neighbors obtained by NREP message. As a result, nodes will be able to predict quality service of flows and forward RREP message if the new flow is locally acceptable. The source node will choose the best path according to the quality of candidate paths. The nodes, existing in selected path, transmit NREP packets to their neighbors in order to verify the status of accepted flows. In this way, all nodes that are affected by the new flow will receive updated information about channel consumption.

The Create_Safe_Path() module cooperates with a module called "Admission Control" module. The latter is used to decide whether a flow should be transmitted over the network or not. It avoids admitting flows more than network capacity. "Admission Control" module needs to estimate exactly channel consumption and to predict quality of the flows (such as operational power and transmission latency) to perform its tasks. It is done by using Model-Based Resource Prediction (MBRP) resource estimation mechanism. MBRP Model calculates operational power and latency of flows by sampling behavior of nodes' back-off. The MBPR Model tries to predict the quality of the flows for both progressing and new traffics, so that an appropriate decision could be made regarding acceptance or rejection of new flow based on quality control and management policy. The input of decomposition model in MBPR is a set of flows such as $A = \{a_1, a_2, ..., a_s\}$ in the network where s is the number of priority classes supported by the system and a_i denotes the number of flows inside ith class. The output of this model is calculation of average delay or operational power for each flow. To address issues such as hidden terminals and unexpected collisions during the run time, MBRP Model utilizes the results of flow progress measurements as feedback. It is worth mentioning that estimation function is analyzed and executed locally. This estimation function located in MAC layer, more properly controls the behavior of packet scheduling and provides realistic (precise) prediction.

Calculation of back-off times with regard to networks status: An algorithm which manages and controls access of nodes to common channel plays a prominent role in regulating smart transmission of flows. It in turn, increases the QoS provided for existing nodes. In this study, the scheduling algorithm, utilized for management and control of node access to channel is based on IEEE802.11 algorithm. There are a variety of algorithms providing various services based on IEEE802.11 algorithm. They consider diverse values of Arbitrary Inter-Frame Space (AIFS), CW and back-off rate for different types of traffics. For example, back-off is higher for low priority traffics while it is lower for high priority ones. Employing aforementioned methods, the services provided for existing flows might be distinguished. That is to say, better services could be provided for high priority flows in comparison with low priority ones.

However, investigating this method, one may conclude that all these parameters are considered static. Static parameters lead to lack of adaptation to network traffic and results in decrease of efficiency. For instance, consider that the configuration assigns small back-off time to high priority traffics and large back-off time to low priority ones. In this case, even if there is no high priority traffics, the low priority traffics must tolerate longer service time. Moreover, a small static value for high priority traffics increases collision and back-off time when several high priority flows compete for accessing to the channel. Thus, it is difficult to find appropriate static values to adjust proportionately parameters of different types of services and network efficiency since these parameters must be changed in accordance with network status. It is impossible to achieve this goal in static conditions. The LWQA has overcome this problem by considering dynamic parameters and noticing the network conditions.

When the network status is included in back-off time calculation, flow transmission would be smarter and the QoS will be improved. If each node detects the network status as saturated, it will set longer back-off time for itself and avoid competing with the flows which are being transmitted. These nodes allow current flows to utilize network resources by assuming longer back-off time and as a result, they avoid probable collisions. On the other hand, a node will choose shorter back-off time if a node detects the network status as uncongested. It utilizes network resources; consequently, it prevents their waste. To include network status in back-off time calculations, R_{col} parameter is added to IEEE802.11 Eq. 1:

$$\text{Back-off} = \text{Rand}\,[0, (2^{r} + R_{col} \times \text{pri}) \times CW] \times \text{Slot\_Time} \quad (1)$$

where, R_{col} parameter denotes the number of collisions occurred between two successful transmissions in one node, pri and r are constant values chosen according to

the priority level of data packets. As it is clear, when the network is saturated both the number of collisions and R_{col} value increase. Any increase in R_{col}, increases the back-off time of the flows. Consequently, the number of contentions in saturated network, decreases and the QoS is maintained at an acceptable level. Conversely, when the number of existing flows is small, the values of both R_{col} and back-off time decrease as a result, it maintains the efficiency of the system in a high level. Considering the Eq. 1, back-off value increases or decreases linearly, i.e., all flows of the same type are treated in the same way. To solve this problem, the current status of each flow must be considered while calculating the back-off time values as it has been explained below.

QoS support using dynamic adjustment of CW: During contention of the nodes for accessing the channel, the node with the least back-off value wins. According to Eq. 1 when CW decreases, back-off value also decreases and the node will have more chance to access the channel. In the example mentioned previously, the flow with longer path is supposed to need less CW value and select shorter back-off time relative to second flow. Since, the requirements of existing flows are different with respect to the type of flows, various methods should be employed to manage CW value.

For this purpose, the existing flows are divided into three groups; delay sensitive flows, bandwidth sensitive flows and best-effort flows. Delay sensitive flows which are mostly conversational, need their data packets to reach the destination in a specific time. Bandwidth sensitive flows need operational power, best-effort flows are resistant to changes in bandwidth and delay. They do not require a specific QoS. In the LWQA, queue data structure is utilized to control and manage CW of flows. It is performed in each node, completely local and without imposing any overload. For each non best-effort flow, one queue is generated (one queue for one flow) and for all best-effort flows passing one node just one queue is formed (Fig. 3).

The type of service that a packet will be receiving is marked in a packet header. It may be mentioned that the "DS" field in IPV4 and "TOS" field in IPV6 are used for the same purpose. Consequently, each node investigates the packet as soon as it receives a data packet. If the received packet is type of non-best-effort, the node puts it in the associated queue. Otherwise, the packet is inserted in the one existing queue which is dedicated to best-effort flows. It is noticeable that these queues are exploited to calculate CW value associated with each flow. Considering different requirements of each mentioned group, a specific CW calculation method is designed for each of them.

CW calculation method for delay sensitive flows: The main parameter regarding delay sensitive flows is the amount of time needed for a data packet to reach destination node (end-to-end delay); in other words, delay sensitive flows must pass through their path in a determined time span; otherwise, it is said that they did not reach their desired QoS. To meet the requirements of these flows, the number of nodes (m) between source and destination should be known. Achieving this value and dividing end-to-end delay (d) by m, the time which is allowed for each data packet to wait in each node is derived. Due to the utilization of the AODV routing algorithm, the number of nodes (m) is known. It is clear that each node is allowed to maintain data packet for d/m seconds; hence, it is called "allowable delay time". Thus, each node tries to limit delay time to the allowable delay time. As a result, sum of delays in all nodes inside the path will be less than d. It is noteworthy that the "allowable delay time" value is carried by a number of initial packets of each flow; therefore, all nodes in the path will be informed about "allowable delay time" value. Then, each node locally calculates CW and tries to maintain delay less than the limit.

With attention to Eq. 2 if the delay of a node exceeds the allowable value for a flow, the CW value decreases. Subsequently, the back-off time decreases. Afterwards, the desired node will take the control of the channel and sends the flow. On the other hand, if the delay is less than allowable value, the CW increases. As a result, back-off time increases and the flow stops competing. Hence, network resources would be free and

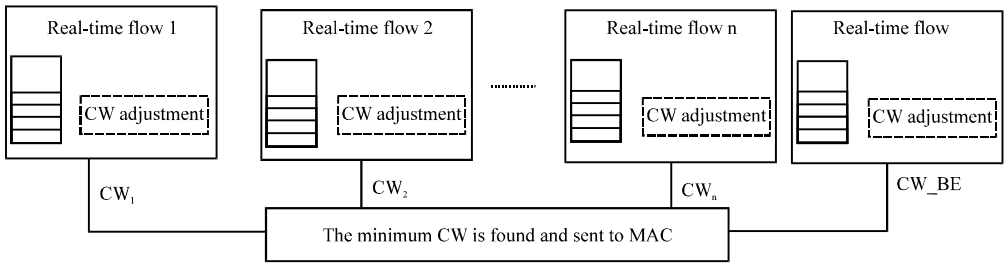

Fig. 3: Queues in the nodes

other flows utilize the resources. Presented formulation for CW calculation associated with delay sensitive flows is as follows. Each node executes the undergoing algorithm alternatively and updates CW of such flows:

$$CW^{(n+1)} = CW^{(n)} \times \left(1 + a\frac{d/m - D^{(n)}}{d/m}\right) \quad (2)$$

Where:

n = The nth update
D = A real packet delay in the node
a = A small positive value which experimentally is considered to be 0.1 in this study

This Eq. 2 moderates the CW value of delay-sensitive flows so that the delay of a packet in each node would be less than the allowable delay value.

CW calculation method for bandwidth sensitive flows: Operational power is the main factor of bandwidth sensitive flows. These flows need in a node, the input rate of packets to be equal to the output rate. According to the queueing theory, the operational power of a flow might be maintained constant by the length of the queue. In other words, constant length of the queue demonstrates that the input and output rate of the flow are equal. It means, required operational power is met. Equation 3 is utilized to calculate CW for bandwidth sensitive flows. The major goal of this equation is to maintain a constant length for the queue and to provide operational power:

$$CW^{(n+1)} = CW^{(n)} + \beta \times (q - Q^{(n)}) \quad (3)$$

Where:

n = The nth update
q = The threshold value of queue length
Q = The real length of the queue
β = A positive value

If the real length of the queue is larger than the threshold, the desired QoS is not fulfilled. Therefore, this equation reduces CW which in turn, increases packet transmission. It compensates for previous shortcomings. In contrast, when the length of the queue is less than the threshold, it increases CW. As a result, the back-off time would increase and packets' transmission rate would decrease. When the length of the queue varies in the vicinity of the threshold value (q), the average of flow's operational power is roughly equal to required operational power. Regarding the guidelines given by Floyd and Jacobson (1993), q was set to be equal to 5 in this study.

CW calculation method for best-effort flows: Best-effort flows are not sensitive to level of provided services. The CW values of this flows are regulated to avoid congestion of these packets in the network. Moreover, it guarantees that non best-effort flows are properly served. While sending such flows, the network status is considered. If more important flows are passing through the network, the best-effort flow will wait. Nevertheless, the waiting time should not be so much that these flows are accumulated and become saturated. For this purpose, the CW of best-effort flows is calculated using the undergoing (Eq. 4):

$$CW^{(n+1)} = CW^{(n)} \times (1 + \gamma \times (f - F^{(n)})) \quad (4)$$

Where:

n = The nth update
f = The congestion threshold value when the channel is free
F = The real free time of the channel
γ = A positive value

The real free time of the channel is the average time between 2 time intervals through which the channel is busy. In this equation, F would be smaller than f when the real-time flows are transmitted.

Later on pseudo-codes related to the packet forwarding, packet receiving, CW calculation and back-off computation will be discussed. When a node receives a packet do the following:

```
Receive_Packet(P)
{
  If (TypeOf(P) = 'Best-Effort') then
  If (there is no queue for Best-Effort flow) then
          Create a queue for Best-effort flow;
  Else if (p is the 1th packet of non B_E flow) then
            Create a queue for this flow;
  Add packet in specific queue;
}
When a node want to send a packet do as following:
Packet_Send()
{
  Indicated which flow the smallest CW relates to.
  If (TypeOf(flow) = 'Real-Time') then
      Find_Fix_Routers();
  Remove a packet from queue;
  Send packet;
  Update queue pointers;
}

Each of nodes performs following instructions to calculating CW for each
flow.
Calc_CW ()
{
  If (TypeOf(flow) = 'delay-sensitive' then

                CW^(n+1) CW^(n) × (1 + a (d/m - D^(n)) / d/m)

  elseIf TypeOf(flow) = 'bandwidth-sensitive' then

                CW^(n+1) = CW^(n) + β(q - Q^(n))

  elseIf TypeOf(flow) = 'best-effort' then
```

$$CW^{(n+1)} = CW^{(n)} \times (1 + \gamma(f - F^{(n)}))$$

}

The proposed algorithm uses the following formula to compute the back-off related to each node:

Back-off_Time()

{

Get minimum $CW_{(CW_{min})}$ from network layer.

Calculate Back-off time according to Back-off = Rand [0, $(2^{r}+R_{col})\times CW_{min}]\times$Slot_Time

}

RESULTS AND DISCUSSION

Model validation: In this study, we study the behavior of a single station with a Markov Model and we obtain the stationary probability π that the station transmits a packet in a generic (i.e., randomly chosen) slot time. Bianchi uses a two-dimensional Markov chain of m+1 back-off stages in which each stage represents the back-off time counter of a node (Fig. 4). A transition takes place upon collision and successful transmission to a "higher" stage (e.g., from stage i-1 to stage i in Fig. 5) and to the lowest stage (i.e., stage 0), respectively.

Each state of this bi-dimensional Markov process is represented by {s(t), b(t)} where b(t) is the stochastic process representing the back-off time counter for a given station and s(t) is the stochastic process representing the back-off stage (0, 1, ..., m) of the station at time t. This model assumes that in each transmission attempt, each packet collides with constant and independent probability p. In other words, p is the probability that in a slot time at least one of the N-1 remaining stations transmits as well. If at steady state each remaining station transmits a packet with probability π, p can be written as:

$$p = 1-(1-\pi)^{N-1} \quad (5)$$

Let:

$$b_{i,k} = \lim_{t\to\infty} P\{s(t) = i, b(t) = k\}, i \in (0, m), k \in (0, CW_i - 1)$$

be the stationary distribution of the chain. A transmission occurs when the back-off time counter is equal to zero. Thus, we can write the probability that a station transmits in a randomly chosen slot time as:

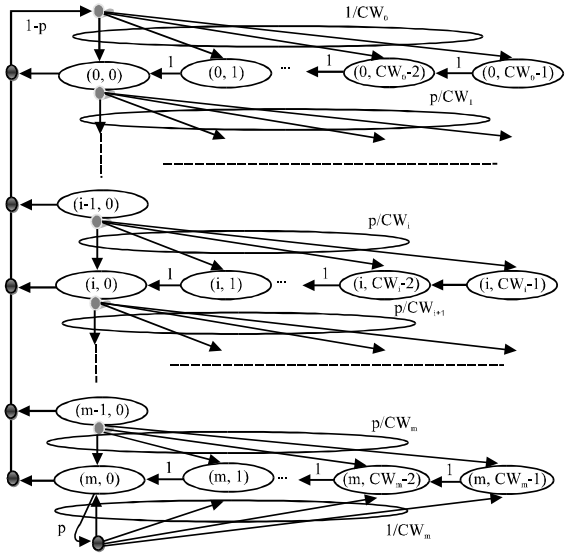

Fig. 4: Markov chain model of back-off window size

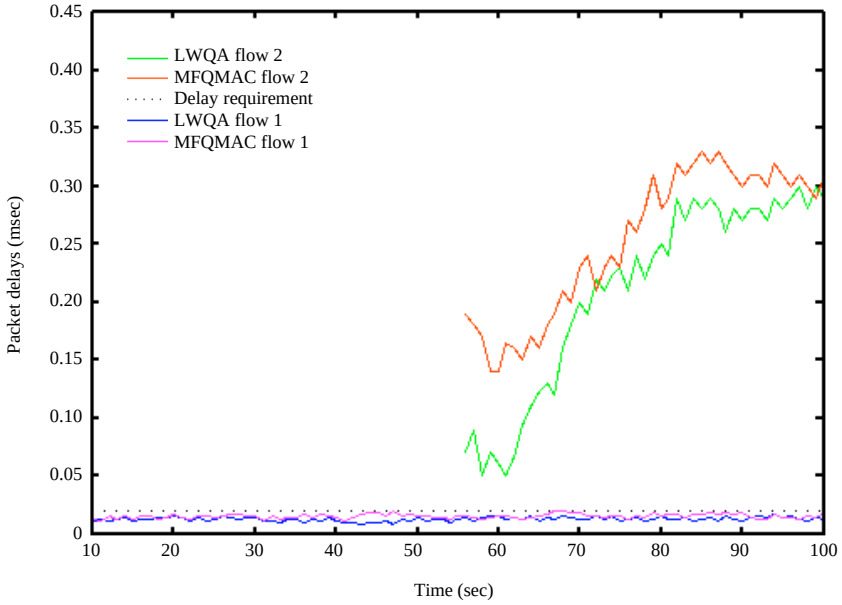

Fig. 5: Delay of data packets

$$\pi = \sum_{i=0}^{m} b_{i,0} \tag{6}$$

For the above Markov chain, it is easy to obtain a closed-form solution for $b_{i,0}$ as a function of p. First, we can write the stationary distribution of the chain for $b_{i,0}$, $b_{m,0}$ and $b_{i,k}$:

$$\begin{aligned} b_{i,0} &= p^i b_{0,0}, \qquad 0 < i < m \\ b_{m,0} &= \frac{p^m}{1-p} b_{0,0} \\ b_{i,k} &= \frac{CW_i - k}{CW_i} b_{i,0}, \; 0 \le i \le m, \; 0 < k < CW_i - 1 \end{aligned} \tag{7}$$

The first and second expressions in Eq. 7 account from the fact that $b_{i-1,0} \times p = b_{i,0}$ for $0<i<m$ and $b_{m-1,0} \times p = (1-p) b_{m,0}$. The 3rd equation can be obtained considering the fact that $\sum_{i=0}^{m} b_{i,0} = b_{0,0}/1-p$ and taking the chain regularities into account (for $k \in (1, CW_i-1)$) that is:

$$b_{i,k} = \frac{CW_i - k}{CW_i} \times \begin{cases} (1-p)\sum_{j=0}^{m} b_{j,0} & i = 0 \\ p \times b_{i-1,0} & 0 < i < m \\ p \times (b_{m-1,0} + b_{m,0}) & i = m \end{cases} \tag{8}$$

By imposing the normalization condition and considering (Eq. 7), we can obtain $b_{0,0}$ as function of p:

$$\begin{aligned} 1 &= \sum_{i=0}^{m} \sum_{k=0}^{CW_i - 1} b_{i,k} = \sum_{i=0}^{m} b_{i,0} \sum_{k=0}^{CW_i - 1} \frac{CW_i - k}{CW_i} \\ &= \sum_{i=0}^{m} b_{i,0} \frac{CW_i + 1}{2} = \sum_{i=0}^{m} b_{i,0} \frac{2^i W_{min} + 1}{2} \\ &= b_{0,0} \frac{W_{min} + 1}{2} + \sum_{i=0}^{m-1} \left(b_{0,0} p^i \left(\frac{2^i W_{min} + 1}{2} \right) \right) + \\ &\quad \left(\frac{b_{0,0} p^m}{1-p} \right) \left(\frac{2^m W_{min} + 1}{2} \right) \\ &= \frac{b_{0,0}}{2} \left[W_{min} + 1 + \sum_{i=0}^{m-1} ((2p)^i W_{min} + p^i) + \frac{p^m}{1-p} (2^m W_{min} + 1) \right] \\ &= \frac{b_{0,0}}{2} \left[W_{min} \left(\sum_{i=0}^{m-1} (2p)^i + \frac{(2p)^m}{1-p} \right) + \frac{1}{1-p} \right] \end{aligned} \tag{9}$$

Thus, $b_{0,0}$ can be written as:

$$b_{0,0} = \frac{2(1-2p)(1-p)}{(1-2p)(W_{min} + 1) + pW_{min}(1-(2p)^m)} \tag{10}$$

Finally, considering (Eq. 6, 7 and 10), the channel access probability π of a node is derived as a function of the number of back-off stage levels m, the minimum contention window value W_{min} and the collision probability p:

$$\begin{aligned} \pi &= \sum_{i=0}^{m} b_{i,0} = \frac{b_{0,0}}{1-p} \\ &= \frac{2(1-2p)}{(1-2p)(W_{min} + 1) + pW_{min}(1-(2p)^m)} \\ &= \frac{2}{1 + W_{min} + pW_{min} \sum_{k=0}^{m-1} (2p)^k} \end{aligned} \tag{11}$$

Considering Eq. 11, how a node obtains a channel and transmits a flow depends upon the rate of collision and CW in each node. That is to say, any node that faces less collision and has the smallest CW obtains the channel with high probability and embarks upon the transmission of its flows. For this reason, nodes can regulate the CWs related to their own flows and provide the desirable QoS. In the proposed, the CW related to the flows is regulated by means of Eq. 2-4. These algorithms are regulated such that they will increase its CW value quickly and provide other flows with the resources existing in the system if a flow obtains a resource more than the required resource at a time and obtains a QoS higher than the desirable QoS. Consequently, other flows will not face any limitations in obtaining resources. On other hand, any node which does not obtain it required resources and QoS at a point in time make much efforts to obtain the resources and compensate for the damages by decreasing its own CW and acquiring much back-off in order to obtain its desirable QoS.

Therefore, it is seen that the algorithm proposed in this study shows a correct function in different conditions in this algorithm, the flows help each other under some circumstances besides quarreling with each other for obtaining resources in order for all the flows in the network to obtaining required QoS.

Evaluation of LWQA: To evaluate the performance of LWQA in supporting QoS, we compare it with MFQMAC using the Network Simulator Version 2 (NS-2). In our simulations, the routing algorithm, channel bandwidth and nodes movement rate were AODV, 11 Mbps and 0-3 $msec^{-1}$, respectively. The evaluations demonstrate acceptable performance of LWQA in improving QoS for network users. Following Table 1 presents the parameters used in our simulations.

Evaluation of LWQA and MFQMAC: To prove the ability of proposed algorithm to properly schedule flows, 2 contending flows were simulated. They had to reach the destination node after 6 nodes. The network area is considered to be 500×500 m. The first flow is a delay sensitive one which should reach the destination node in 20 msec. The second flow is a bandwidth sensitive flow transmitted at t = 55 sec. Both flows have a transmission rate equal to 30 512 byte packets per second. Figure 5 depicts the delay of data packets by these algorithms.

As it can be seen, till 55 sec both algorithms are able to maintain the delay of flows less than their required time (20 msec). Even at t = 55 sec when the second flow starts, the algorithms are capable of controlling the delay of delay sensitive flows. To do so, they increase the delay of second flow which is bandwidth sensitive. As the second flow is not sensitive to delay, it is considered as an acceptable attempt. The figure illustrates that the LWQA, using the low speed nodes, provides more precise control. The ability of LWQA could be seen by increasing the number of nodes and flows in the network. Hence, in the next simulation; the number of nodes, flows and network scale are increased to assess the algorithms.

Table 1: Simulation parameters

Parameters	Values
α	0.1
β	1
f	1 msec
r	0.1
q	5 packets
pri for real-time flows	0.5
pri for best-effort flows	1
Update interval of CW	0.1 sec

Evaluation of LWQA and MFQMAC in large scale ad hoc network with increase in number of nodes and flows: In this study, we evaluate LWQA and MFQMAC's ability to keep QoS guarantees to delay-sensitive flows. The simulation area is considered to be 1000×1000 m. In this simulation, 8 delay-sensitive, 8 bandwidth-sensitive and 8 best-effort flows are utilized which start in the 1st 115 sec of the simulation. These flows generate 50 512 byte packets/s. Each delay-sensitive flow should be served in order to reach the destination in 100 msec. The source and destination of the flows are randomly chosen from 150 nodes inside the network. The number of nodes that each flow needs to pass is considered as a random value between 1 and 7 diametric. Figure 6 shows the average delay of delay-sensitive flows. The delay requirement of these flows is indicated by the dotted line in this Fig. 6. Both algorithms have kept the delay of flows less than the required value; nevertheless, LWQA demonstrates better performance because of 2 reasons. First, it utilizes static and low speed nodes during routing non-best-effort flows. Second, it considers the flows' statuses as well as current status of the network in calculating back-off time. It can be assumed that LWQA is able to manage networks with larger number of nodes. To verify this assumption, these algorithms were evaluated once more.

Evaluation of LWQA and MFQMAC in large scale ad hoc network with extra increase in number of nodes and flows: The area in last experiment is 2000×2000 m and

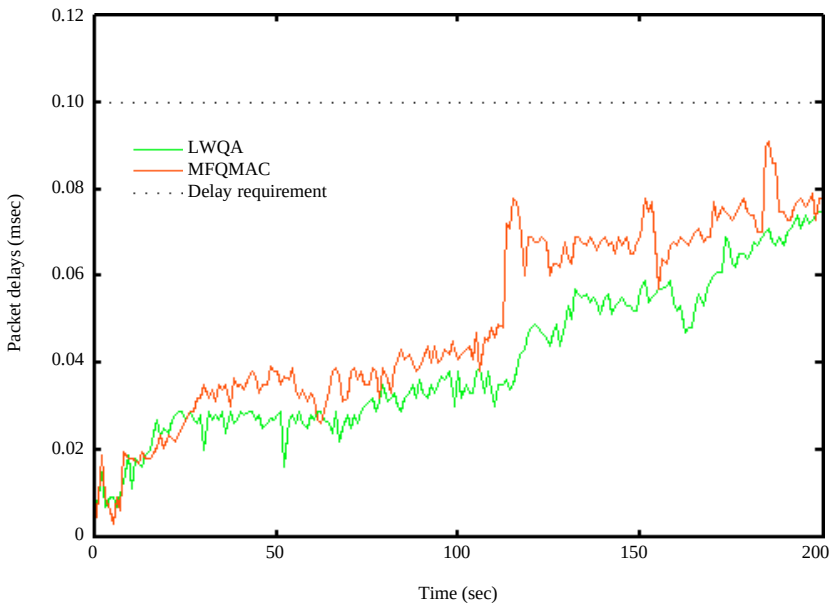

Fig. 6: Average delay of delay-sensitive flows

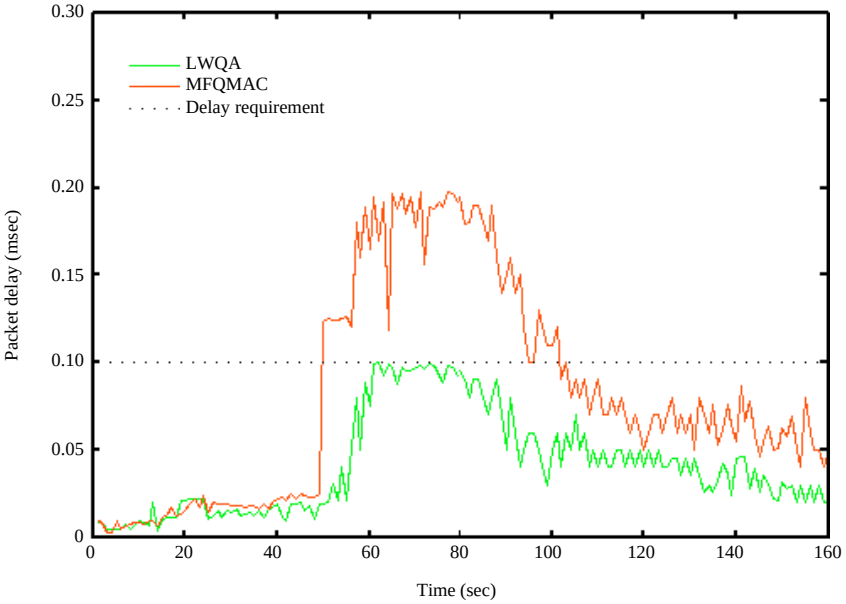

Fig. 7: Average delay of delay-sensitive flows

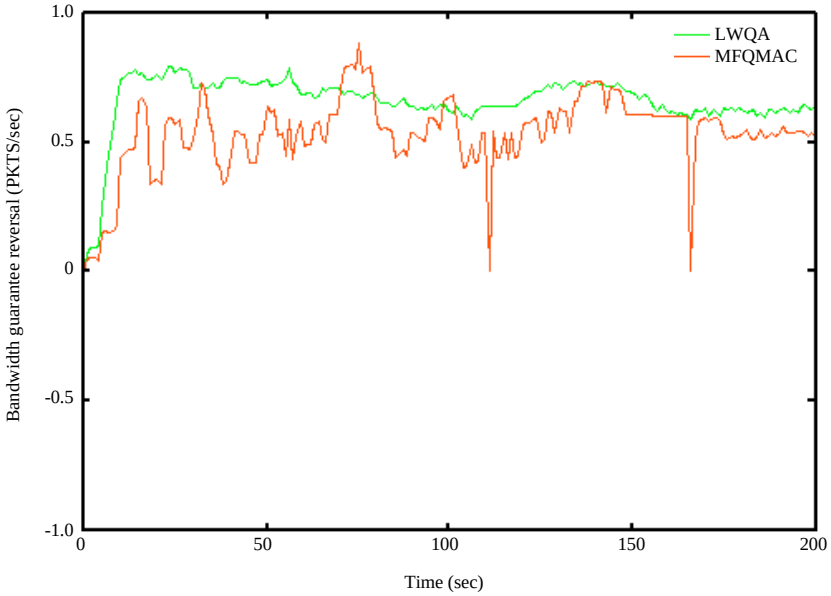

Fig. 8: Bandwidth guarantees to bandwidth-sensitive flows

180 nodes are randomly distributed. The numbers of delay-sensitive, bandwidth-sensitive and best-effort flows are 20, 25 and 25, respectively. The source and destination nodes, chosen randomly from existing nodes, communicate with each other. As the goal of this experiment is to investigate the behavior of algorithms in the networks with large number of nodes, the packet size and rate of the flows are considered to be like the previous experiment. In the 1st 50 sec of this simulation, 4 real-time, 4 best-effort and 4 bandwidth-sensitive flows were transmitted between nodes of the network. Figure 7 and 8 show the average delay of delay-sensitive flows and the violation of bandwidth guarantees to bandwidth-sensitive ones, respectively. As shown, LWQA controls the delay of the admitted delay-sensitive flows below their required time (100 msec) and shows no violations to the bandwidth guarantees. On the other hand, regarding MFQMAC, when other residual flows are

transmitted within 50 and 80 sec of the stimulation and network load increases, admits too many flows in a way that both delay and bandwidth are degraded. Nevertheless, LWQA keeps QoS guarantees for these flows. It achieves this goal by considering admission control, network status and current status of flows in back-of time calculation also it is the case by using static and low speed nodes during routing process.

In this simulation at time 80 sec of the simulation, 4 delay-sensitive, 4 bandwidth-sensitive and 4 best-effort flows come to end, leading to a decrease in the number of existing flows. As a result, MFQMAC gradually obtains its control over the flows and provides them with the required QoS. But the time needed for MFQMAC to return to the normal condition is more than that the time needed by LWQA. Therefore, LWQA provides more stable packet delay and operational power than MFQMAC.

CONCLUSION

The proposed algorithm aims to improve QoS in ad hoc networks. Several algorithms have been introduced but the proposed one is advantageous from different perspectives. Firstly, it does not need resource reservation to provide QoS. It classifies the flows and provides different services for them in accordance with their requests. This is the common characteristic of various algorithms; nonetheless, the novelty of LWQA is the ability to provide different types of services for flows which are of the same type. In most of models, supporting QoS, all flows compete to obtain their needed QoS. In the LWQA, a flow avoids contending other flows and tries to improve their QoS if a flow achieves required QoS by using Eq. 2-4. Finally, by using Create_Safe_Path() module, this algorithm distinguishes between fast and slow nodes. It utilizes slow nodes to send important data packets. The simulation results revealed the ability of LWQA algorithm to improve the QoS for ad hoc networks.

REFERENCES

Abbas, C.J.B., L.J.G. Villalba and A.L.S. Orozco, 2012. A distributed QoS mechanism for ad hoc network. Int. J. Ad Hoc Ubiquitous Comput., 11: 25-33.

Abdullah, A., N. Ramli, A. Muhammed and M.N. Derahman, 2009. A performance study of routing protocols for mobile grid environment. J. Inf. Commun. Technol., 8: 41-54.

Bouhouche, H. and S.G. El-Fatmi, 2010. A QoS-based resources reservation mechanism for ad hoc networks. Int. J. Comput. Appl., 6: 1655-1658.

Budyal, V. and S.S. Manvi, 2013. Intelligent agent based delay aware QoS unicast routing in mobile ad hoc networks. Int. J. Multimedia Ubiquitous Eng., 8: 11-28.

Diaz, J.R., J. Lloret, J.M. Jimenez and S. Sendra, 2014. MWAHCA: A multimedia wireless ad hoc cluster architecture. Sci. World J.

Elizarraras, O., M. Panduro, A.L. Mendez and A. Reyna, 2014. MAC protocol for ad hoc networks using a genetic algorithm. Sci. World J., Vol. 2014.

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

Ghazali, O. and N. Harun, 2012. Implementation and validation of an adaptive FEC mechanism for video transmission. J. Inf. Commun. Technol.,

Mangold, S., S. Choi, P. May, O. Klein, G. Hiertz and L. Stibor, 2002. IEEE 802.11e wireless lan for quality of service. Proc. Eur. Wireless, 1: 32-39.

Perkins, C. and E. Belding-Royer, 2003. Ad hoc on-demand distance vector (AODV) routing-request for comments-RFC3561. Proceedings of the WMCSA'99, February 25-26, 2003, New Orleans.

Preveze, B. and A. Safak, 2012. Effects of routing algorithms on novel throughput improvement of mobile ad hoc networks. Turk. J. Elec. Eng. Comput. Sci., 20: 507-522.

Seth, D.D., S. Patnaik and S. Pal, 2013. MFQMAC: A faired and QoS assured MAC protocol for multihop adhoc network. Int. J. Comput. Appl., 66: 28-34.

Vijayalakshmi, A. and P. Ramamoorthy, 2013. Performance analysis of genetic algorithm based unicast and multicast routing optimization model for mobile ad hoc networks. J. Theor. Appl. Inf. Technol., 56: 183-192.

Wang, P., H. Jiang and W. Zhuang, 2008. A new MAC scheme supporting voice/data traffic in wireless ad hoc networks. Mobile Comput. IEEE Trans., 7: 1491-1503.

Zhang, X.M., Y. Zhang, F. Yan and A.V. Vasilakos, 2015. Interference-based topology control algorithm for delay-constrained mobile ad hoc networks. Mobile Comput. IEEE Trans., 14: 742-754.