**Achieving Application Level Fairness Using Bayesian Classifier**

**ABSTRACT**

Various fairness technique proposed by researchers for wired networks can be applied for ad hoc wireless network. The end-to-end fairness in an ad hoc wireless network is a challenging task compared to wired networks, which need to be addressed. Most of the traffic in an ad hoc network are transport layer flows and thus the fairness of transport layer flows has attracted the interest of the researchers. Recently, the network architecture has changed rapidly in modern applications and has imposed priority for heterogeneous applications. In some cases, the priorities of the application need to change in the middle of the communication. However, the major problem is that the packets of different applications are not served at the destination by their expected priority ratio even if the priority does not change. Considering the above problem, we presents a fair packet scheduling policy which collects information from each individual flow according to its priority and automatically classify to different application using Bayesian classifier. The proposed method has been implemented in Java Weka for different numbers of applications. The result shows that the proposed protocol achieves significant improvement in terms of fairness and throughput in comparison to existing protocols.

**Table of Contents**

Certificate

Acknowledgement

Abstract

Chapter 1: **Introduction**

1.1 Fairness approach

1.2 Node-based fairness

1.3 Flow-based fairness

1.4 Queuing models

Chapter 2: **Literature Survey**

Chapter 3: **Proposed Approach**

3.1 System Architecture

3.2 Overview of Bayesian classifier

Chapter 4: **Classification and Scheduling Algorithms**

4.1 Calculation of threshold variable

4.2 Network Scenario

4.3 Scheduling Algorithm

Chapter 5: **Congestion control mechanism**

5.1 Congestion detection

Chapter 6**: Experimental Results**

6.1 Performance evaluation parameter & results

Chapter 7**: Conclusion**

**References**

**CHAPTER-1**

**INTRODUCTION**

Ad-hoc networks are complex distributed systems that consist of wireless mobile nodes that can freely and dynamically self-organize. In this way they form arbitrary and temporary “ad-hoc” networks topologies, allowing devices to seamlessly interconnect in areas with no pre-existing infrastructure. In the emerging field of current researches, ad hoc and sensor networks have an important role because of its broad areas of applications-like climate forecasting, emergency search, agriculture, rescue operations, military control, command operations in a battlefield etc.

However, there exist some important difficulties in order to achieve a smooth communication with ad hoc networks. The Transmission Control Protocol (TCP) over IEEE 802.11 and the media access control is designed for wired network and found to be not suitable for multi hop ad hoc wireless network. The properties of these protocols have demonstrated serious concerns and thus a large number of research studies have highlighted several research issues of TCP in ad hoc networks. The most important technical factor is the reliability which is obvious due to the absence of a fixed infrastructure. The ad hoc network is dynamic in nature due to various parameters such as transmission of control packets, multi hop nature of forwarding packets, changes in source and destination nodes, changes in the routing path influences determining throughput and fairness among the concurrent flows. In order to avail a smooth wireless communication, the most important work is to consider the reliability and timeliness. When nodes are connected through ad hoc technology, the major challenge is to consider fairness among all nodes. In addition to this, when the nodes are subject to work with multiple applications, then this challenge is again combined with the fairness among all flows.

The existing transport layer protocols are not able to provide fairness of the different nodes and applications simultaneously. Thus, the current research work focuses on fairness among heterogeneous flows with dynamic change of priorities along with congestion control. Relating to priority, many techniques have been proposed to achieve fairness. Whereas, various issues are addressed that include fairness among applications and nodes. Furthermore, fairness among heterogeneous applications includes flows with different priority and equal priority, i.e., the flows should transmit data according to their priority. The fairness models work towards the fairness with respect to nodes, which defines that each node should get equal opportunity to communicate. Likewise, the researchers have put an attempt to consider fairness for applications. The most important issue is Fairness among all applications which describes, every node and application should get equal chances to send packet according to their priority. Sometimes priority of different nodes or applications may change dynamically due to the environment which is measured and analyzed in real time applications by destination station. Hence there is a need for a scheme to transmit packets from the entire nodes equivalent to their need, set by their application and priority. The communication of a network is said to be fair when all applications can send equal number of packets to the destination unless they pertain different priority. Hence, receiving data packets according to the priority of the application is needful which is called as application level fairness. Here, we define the fairness of communication as the ratio of packet types received by the sink node should be equivalent to their priority. Thus, the current research work focuses on fairness among heterogeneous flows with dynamic change of priorities along with congestion control.

In this report, we suggest Bayesian classifier based solution to maintain fairness among applications. This technique describes that every packet has its own identity to represent its type of application to go for queuing in the limited buffer of the node. The buffer space is logically partitioned into the number of applications of the network (interface queues) and allocates queue lengths according to their priority. The Bayesian classifier which classifies the packets by their application-ID stores packets in the queue according to the availability of memory in their respective queues. Likewise, the packet dispatch scheduler also schedules the packets to be transmitted next by their turn followed by availability. In this way, the proposed technique claims fairness for each type of applications. As packets are buffered in different queues according to their priorities and dispatched through the scheduler, that implements weighted round robin scheduling for fairness among different flows. Hence the number of packets received at the destination node should be according to their priority also the bandwidth used by different applications is according to their designated ratio.

The important approaches followed to achieve fairness in existing literatures are sketched. Additionally, issues regarding cost analysis of nodes and queue models are discussed for better understanding.

**Fairness approach**

Fairness is broadly classified into two types: priority associated with nodes or regions and priority associated with flow types. However, performance evaluation of fairness is basically of two types: fairness based on throughput and time stamp. Throughput-based fairness refers that service rate of packets should be fair in terms of bandwidth consumed/packets received. When bandwidth consumption is measured with respect to the nodes, it is called as ‘node-based fairness’ and when it measured with respect to the applications, it is called as ‘application-based fairness’. Similarly, time stamp-based fairness refers that the average end-to-end delay of higher prioritized packets should be less as compared to packets having lower priority.

**Node-based fairness:**

Node-based fairness is defined as any node of the network should share equal bandwidth as compared to other nodes. CCF has given a solution for the problem by assigning the bandwidth to its predecessors reciprocally with their sub-tree size. As described in CCF, the leaf node has sub-tree size 1. Hence, Figure 1, *Q* has 3 child nodes (*R*, *M* and *L*). Where *R* and *L* have sub-tree size 3 and *M* has 9. So, *R* can share 3/(3 + 9 + 3 + 1) times of bandwidth of *Q* to send data, but *M* can share 9/(3 + 9 + 3 + 1) times of bandwidth of *Q* to send data.



Figure 1: Node deployment in the network

**Flow-based fairness:**

It defines that packets of different flows in the network should get equal opportunities to be transmitted at any router if they have equal priorities, otherwise they should be transmitted equivalently with their priority ratio, i.e., Where *ρ*1 and *ρ*2 represent the priority of flow 1 and flow 2, respectively, whereas *Sink*1 and *Sink*2 represent the amount of data received at *Sink* node by flow 1 and flow 2, respectively. The researchers have proposed their models to provide fairness by allocating different queues to different flows. However, they increase flow rate for the application having higher priority. Researchers also have suggested that in every short span of time (*t*1 to *t*2), the network should support fairness with respect to flows and applications.

**Queuing models:**

There are different queuing models suggested in literature (Monowar et al. (2012), Wang et al. (2006), Iyer et al. (2005) and Ee and Bajcsy (2004)), designed to manage packet scheduling at any node. According to these models, the packets can be received and transmitted on certain conditions. Some of the well-known queuing models available in the literature are presented below.

1. PCCP places the packets according to their generation position, i.e., the protocol manage two types of queues, one for packets generated locally and another for packets in transit. In such cases, there is no flow-ID or node based monitoring mechanism
2. STCP and PHTCCP store packets in a node according to their flow-ID. That means, the packets having an equal flow-ID are placed in the same queue which can maintain fairness with respect to flows
3. The model implemented in CCF manages queues according to its local receiver node address. However, the intermediate node cannot create a queue for each source node. But, the priority of a node is calculated by sum priority of its sub-tree nodes added by one (Note: each leaf node has sub-tree size 1). In this scenario, despite the type of flow, the packets from a significant node are being stored in a specified queue and one more queue is created for the packets generated locally. In this way, it claims approximately equal number of packets are served from each.

**CHAPTER-2**

**LITERATURE SURVEY**

A number of protocols are designed in the area of ad hoc sensor networks to maintain the fairness in the flow of packets among heterogeneous applications. But, most of the schemes follow first come first serve (FCFS) scheduling policy for intra-queue packet scheduling and dispatch packets from different queues using the round robin technique. In most of the schemes, they are focused on overall delay for the flows and in all schemes, either low priority packets drop due to starvation or the data contents become obsolete as they can’t reach within the required time, and cannot justify proper fairness.

In PCCP (Wang et al., 2006), the scheduler differentiates packets according to their application type by maintaining two queues, one for current node and another for packets from transit traffic. It claims to reduce packet loss and maintain fairness in terms of nodes with the support of multi-path routing. Three techniques as the intelligent congestion detection (ICD), implicit congestion notification (ICN) and priority-based rate adjustment (PRA) are used in this protocol to achieve fairness for each node. But, it cannot provide per flow fairness as it does not differentiate packets of different flows.

In PHTCCP (Monowar et al., 2012), the classifier differentiates the packets according to their application IDs and places in their corresponding queues. The scheduler then decides to dispatch the packets according to their priority using weighted fair queuing (WFQ). Here, the number of queues created is equal to the number of application types, the node deals with. Also, it follows multiple paths to transmit packets and maintains link utilization when some node(s) in a particular route is/are inactive or in sleep mode. Sometimes, the packets from a given source of same application type reach at an intermediate node in different times. Later, the packets those have experienced more transit delay cannot get higher priority to reach at the destination. This particular situation cannot serve for a better reliability.

STCP (Iyer et al., 2005), modifies the TCP/IP packet header format. It is implemented for multiple flows. The packet header contains flow-ID to differentiate packets from different flows. In this protocol, only one queue is maintained, but the priority of the packet was set according to their flow-ID. The flow-ID guides the scheduler to select a packet to dispatch. First in first out (FIFO) queuing technique is used to dispatch packets for same flow-ID that are stored in a given queue. As it applies FIFO to dispatch packets, it cannot achieve weighted fairness to each flow.

Wakuda et al. (2009) have proposed a max–min fairness algorithm for wireless local area networks. They create different queues for each flow and assign their weights in another array known as weight counters. The weight of the queue is determined by the number of packets stored in the packet. However, the above approach is not applicable in ad hoc sensor networks due to limited buffer memory of wireless nodes.

Time-critical event first (TCEF) scheduling policy is used to determine priority of the packet(s) in RT2 (Gungor et al., 2008). One more field is also associated with each packet named ‘Time Elapsed’ like in Monowar et al. (2012). This field signifies the remaining time of deadline to reach at the destination. The priority of the packet is measured using this elapsed time field. Total communication is differentiated into two types. They are sensor–actor communication to gather information from the sensors and actor–actor communication to manage or reconfigure the nodes for an optimal use of sensors. They have considered higher priority to the packets belongs to actor–actor communication. They focus only on end-to-end delay but do not focus on priority for flows.

Yuan and Duan (2009) have proposed a simple and fair round-robin technique among different flows having different priorities. However, they recognize the priority of the flow as high if the flow exhibits a higher data rate. The scheduler records the packet incoming rate and schedules the packets to be dispatched accordingly. Ramabhadran and Pasquale (2003) have proposed stratified round-robin (STRR) technique to achieve lower end-to end delay irrespective of flows which can be implemented any router by a small change in hardware. However, they have not considered the fairness with respect to the throughput and weighted priority of flows.

**CHAPTER-3**

**PROPOSED APPROACH**

To achieve fairness for each type of flows, we propose a stronger scheduling policy. We also propose the model to update the priority of flows for dynamic changes along with buffer usage and channel utilization-based congestion prevention technique. The protocol describes that every packet has its own identity to represent its type of application to go for queuing in the limited buffer of a node. The buffer space is logically partitioned into the number of applications of the network (interface queues) and allocates queue lengths according to their priority. The classifier which classifies the packets by their application-ID stores packets in the queue according to the availability of memory in their respective queues. Likewise, the packet dispatch scheduler also schedules the packets to be transmitted next by their turn followed by availability. In this way, the proposed protocol claims fairness for each type of applications. Moreover, the model updates scheduling rates when congestion is detected using service time based congestion detection policy and prevents congestion.

**System Architecture:**

Inour proposed method we classify the incoming packet based on their application- ID by accessing their IP header. All incoming packet passes through a classifier which classify the packet in various applications based on their application-ID. If we found new application then assign them into logically divided new queue else assign the packet to existing queue of existing application. All packets which will reside in logically divided queue passes through the fair queue scheduler which schedule the packet based on their priority of application to hardware queue.

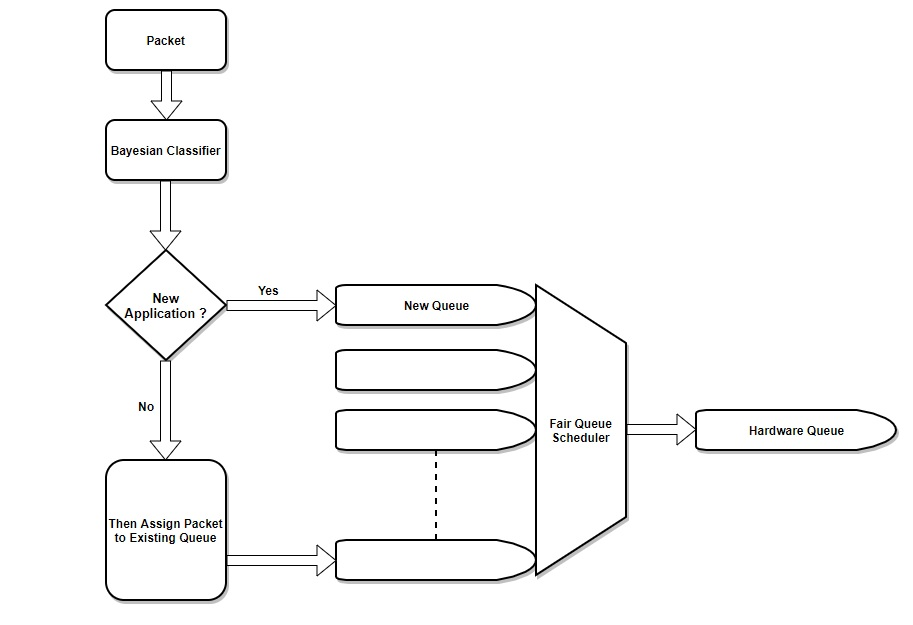
****

Figure 2: System Architecture

**Overview of Bayesian Classifier:**

In heterogeneous flow, it is very difficult to classify packets of different application. For classification of incoming packet in the network we use existing classifier as a Bayesian classifier.

**Bayesian Classifier:**

The Bayesian Classification represents a supervised learning method as well as a statistical method for classification. Bayesian classification provides practical learning algorithms and prior knowledge and observed data can be combined. A Bayesian classifier is based on the idea that the role of a (natural) class is to predict the values of features for members of that class. Examples are grouped in classes because they have common values for the features.

The idea behind a Bayesian classifier is that, if an agent knows the class, it can predict the values of the other features. If it does not know the class, Bayes' rule can be used to predict the class given (some of) the feature values. In a Bayesian classifier, the learning agent builds a probabilistic model of the features and uses that model to predict the classification of a new example.

Bayesian model is a representation of the examples as points in space, mapped so that the examples of the different categories are divided by a clear gap i.e. as wide as possible .In most of classification problem the difficulties and subtleties are associated with modelling the probability distributions effectively. In classification problem the Bayesian classifier minimize the probability of misclassification. As we have mainly focused on dynamic priority associated with the packet, Bayesian classifier gives full advantage of dynamic priority in classification.

**CHAPTER-4**

**CLASSIFICATION AND SCHEDULING ALGORITHMS**

In this proposed model, each node has *N* queues (*Q*1, *Q*2, …, *QN*), i.e., one queue for each type of applications. For the above-assumed scenario, here we considered for different queues for different application. The first queue (*Q*1) stores all packets of the flows (*P*1, *P*2, …, *PN*) , the second queue (*Q*2) stores all packets of the flows of another application, similarly for different queues of application. The threshold variable (*Ki*) is used to allocate the queue length of applications and moderate it when the priorities of application(s) change in dynamic scenarios. The proposed model contributes with designing the algorithms for both classifier and scheduler. The model also maintains its fairness when priority of any flow(s) changes dynamically. The classifier implements the algorithm (shown in Algorithm 1.1) which logically divides the node buffer into *N* queues and assigns different queue lengths reciprocally with the help of priority threshold *Ki*. The packet dispatch scheduler uses a counter-based weighted round-robin policy for packet scheduling which is shown in Algorithm 1.2.

**Algorithm 1.1: Classification**

**Procedure Classification (Packet p)**

For each packet *P*i

*A\_idi* ← application id of flow i

if *A\_idi* belongs to existing class of application

Assign to existing class of application i

else

Assign to new class of application i

**Algorithm 1.1: Classification**

**Procedure enqueue (Packet p)**

i← Application type i

Kicurr ← Current threshold of application i

if Qi Len ≤ Kicurr\*Qmax

enqueue packet to Qi.

else

drop the packet.

**Algorithm 1.2: Scheduling**

**Procedure Scheduling (Queue*turn*, Applicationi)**

*Turn*←The queue form which application of packet supposed to dispatch

For each class of application i

P←dequeue the packet p from Qturn.

if(p!=NULL)

increment the Counteri by 1

if Counteri≥ Kicurr\*Qmax

Counteri←0

if application is lagging

Update the Kicurr using updating equation

Return packet p

else

Dequeue(Q(i+1)%R)

**Calculation of threshold variable (Ki):**

The priority threshold variable logically divides the queues according to the priorities of the applications. The objective of introducing a variable, i.e., priority threshold variable (*K*) is to allocate queue length to each application so that it can achieve fairness. Hence, initially the value of (initial value of *K* for application *i*) is calculated as:

where *N* is the number of applications exist and *ρi* is the priority of the application of type *i*. Let, there are two applications and the priority of the first application (*ρ*1) is three and the priority of the second application (*ρ*2) is two. According to the above formula, the value of is calculated as 3/(3 + 2), i.e., 0.6 and is 2/(3 + 2), i.e., 0.4.

**Network scenarios:**

Basically, two scenarios are considered, i.e., static and dynamic. A different priority model is presented for each scenario. In the static scenario, priority is not changed but in dynamic scenario, priority changes in the middle of the communication. However, when priority changes dynamically the threshold can be updated by two methods as per requirement. Thus, the threshold models are:

1. static threshold model
2. dynamic threshold calculation model
3. threshold calculation by intermediate node
4. threshold calculation by sink node.

**Static threshold model:**

In this model, the threshold value remains unchanged throughout the communication. This model is useful in error less channels and the scenarios where priorities of the applications are not supposed to change in the middle of the execution. Hence, the priority threshold variables of the applications do not need refreshing and any update message is not required to broadcast. Initially, the value of threshold variable is calculated using equation.

**Dynamic threshold model:**

In this model, two techniques are proposed to accommodate the priority threshold value with the required priorities when the priority of application changes dynamically. They are classified based on who is going to update threshold values, i.e., threshold value calculated by sink node and a threshold value calculated by intermediate nodes. The base station triggers the updated priority into the network. Moreover, the base station undergoes for a regular analysis of available information. In the middle of the execution, if the analytical result suggests any change in priority then also it acknowledges to the network.

**Threshold calculation by intermediate node:**

The intermediate nodes calculate priority when it receives an instruction from the user that the priority of any specific application (say temperature) to increase. The user calculates the value of the threshold variable (*Ki*) and broadcasts to its descendant nodes of the network. To balance the sum of total threshold variables, every intermediate node considers the next application (i.e., *i* + 1th application) as lagging and updates both *Ki* and *Ki+1*. It is noticed that the value of *Ki* can be updated only if it is less than 1 (i.e., 0 ≤ *Ki* < 1). The intermediate nodes update priority threshold as follows:

, if application is lagging

=

, if application is leading

**Threshold calculated by Base station:**

In this technique, the base station broadcasts the value of threshold variable explicitly to the downstream nodes. The intermediate nodes update the value of the threshold variable and send to all its downstream nodes. Here, the network faces a very less overhead for message passing and reduces calculation complexity at each wireless node. This is not a difficult task for the base station as it has sufficient resources to calculate any kind of operations according to the need of the user. This model gets initiated when the base station requires change or at predefined time set by the user.

**Scheduling algorithm:**

In this section, two different algorithms are proposed for classifier and scheduler. The algorithms are encoded inside the queuing technique. The classifier receives the packets and stores them in the queues whereas the scheduler schedules the packets to be transmitted next. All intermediate nodes receive the priorities of flows from the network by accessing the IP headers of the packets, which are numerical values and set queue lengths for each application. It is worthwhile to notice that priority threshold variable (*Ki*) plays an important role for logical classification of queue length and packet scheduling. Taking an example for two applications, if the value of *ρ*1 is 3 and *ρ*2 is 2. Let, total queue length is 30 then, according to equation (2) the first queue gets 18 slots and the second queue gets 12 slots to store packets.

In Algorithm 1, the classifier first checks the priority of the received packet by accessing its IP header. When any intermediate node receives a packet, it detects its priority and creates an interface queue for it if not created yet. Thus, the intermediate nodes get to know all the priorities it deals with and calculates their proportional ratio and stores in the variable (*Ki*). According to the value of the priority threshold (*Ki*), the node decides the length of the interface queues in the equal proportion with priority and allows the packets to be stored in their respective queues. In this way, a classifier stores packet in its respective queues only if there exist any vacant slot otherwise simply drops it. On the complexity point of view, the proposed algorithm does not take more time to process a packet at classifier. It can process only in constant times, i.e., *O*(1). Thus, the classifier plays an important role in assigning different queue lengths for each queue and to store the packets in respective queues. The working principle of the scheduler is presented in Algorithm 2. The current priority threshold variable ( ) is used to dispatch packets which ensures that the packets are served in the order of their priority. *Queue\_Turn* and *Counteri* are used to decide the queue pointer which is going to send a packet. It sends a packet from the queue *i* if it is the turn for it and increments *Counteri* by one. The counter reset to zero and *Queue\_Turn* flips to next queue when the counter value reaches its threshold ( × ) otherwise continues to send from the same queue. Thus, the algorithm encoded in scheduler runs in weighted round-robin fashion to provide fairness according to their assigned priority. It can be verified that the algorithm executes in a constant time, i.e., eight units of time, which can be presented as *O*(1). Moreover, it can be noticed that the scheduler can process the algorithm at idle time to minimize the delay. In case, if the turn for a specified queue is active but there does not exist any packet, then it’s lagging counter increases by one and the scheduler transmits a packet from the other queue. However, the counter of that queue does not increase. In the next turn, the application is permitted to send lagging number of packets. When its lagging counter touches a threshold value or the queue becomes empty for a predefined time span it is assumed that the channel/flow perceives error. In this way, the algorithm maintains the fairness and utilizes other queues in absence of packets.

**CHAPTER-5**

**CONGESTION CONTROL MECHANISM**

**Congestion detection:**

In this proposed model, PSR(*i*) denotes packet service ratio to detect congestion level at each node *i*. It is the ratio of the average packet sending rate and the average packet scheduling rate at the transport layer in each node *i*, that is,

PSR(i) = (PSNDRi) / (PSCRi).

Here, PSNDRi also can be presented as (1/ ) where, is packet service time. Packet service time is the time difference between receiving time at the MAC layer and successfully forwarded to the next hop. Thus, is the total time spent by a packet including packet processing time, queuing time, waiting time, and packet transmission time. To obtain PSNDRi, the intermediate node calculates and updates dynamically using linear formula as:

= *w*i × + (1 - ) × last\_packet()

where last \_ packet( ) is the packet service time of thepacket that just has been forwarded and *ws* is a constantwhere 0 < *ws* < 1. The packet service ratio indicates thecongestion level of a node. When it becomes greater than orsame as 1, it shows that the packet receiving rate of a nodeis less than its service rate and indicates less chance ofcongestion. However, when packet service ratio becomesless than 1, it indicates both link level and buffer levelcongestion.

**CHAPTER-6**

**EXPERIMENTAL RESULTS**

Experimental results for all different scenarios which are taken into consideration are presented. The performance of the proposed protocol is evaluated for dynamic scenarios. The relative performance is evaluated with FCFS policy, PCCP policy, ICD policy and PRA policy.

**Performance evaluation parameters & results:**

**Network throughput:** Throughput represents the network productivity during the network operation. The network throughput is represented by the number of packets successfully transmitted from source to destination (*PR*) or bits received during a period of time, i.e.

Throughput =

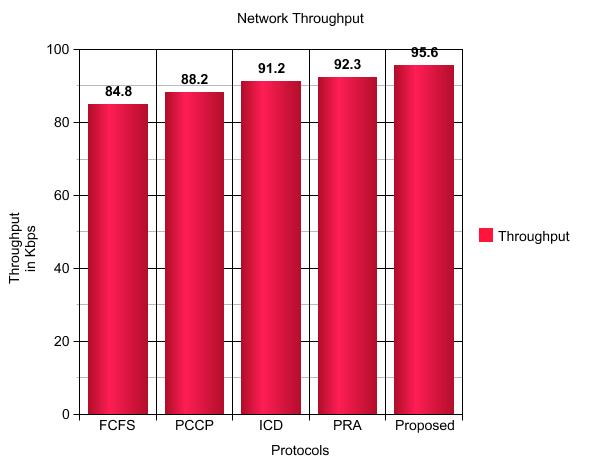


Figure 3: Network Throughput

**Packet success rate (PSR):**

It represents the number of packets received at destination to the number of packets generated (*PS*). However, the packets generated include retransmissions due to packet drop. Sometimes, it is presented in terms of percentage of success rate, i.e.,

PSR (%) = × 100

**Packet drop rate (PDR):**

*PDR* represents the number of packets dropped (*PD*) to the number of packets generated. However, there are many causes of packet drop such that: drop due to interference (*δα*), packet drop due to unavailability of space in the node buffer during buffering (*δβ*), drop due to attenuation of energy during transmission (*δγ*), drop due to fading of signals at the mobility of nodes (*δη*), etc. Also *PDR* can be calculated from *PSR* and other methods, i.e.,

PDR (%) = × 100

*PD = δα + δβ + δγ + δη + δothers*

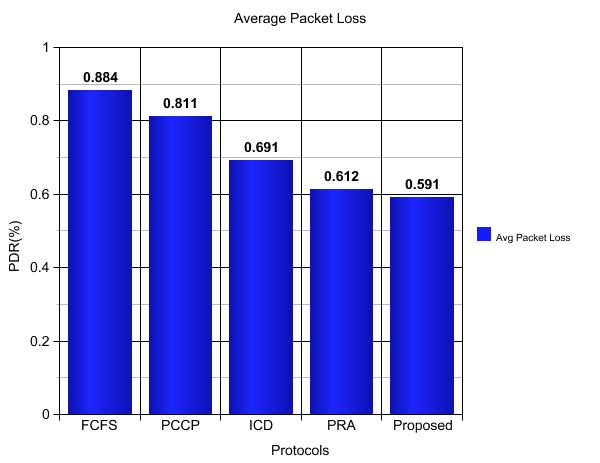
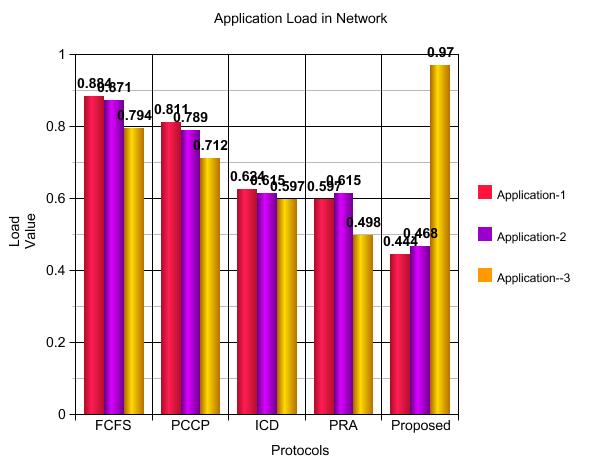
PDR=1 – PSR

Figure 4: Packet Drop Rate

**Application Load in network:**

****Application load can be calculated by the ratio of bytes served in server busy time and total bytes served in total simulation time, it can be given as:

Application Load **=**

Figure 5: Application Load in Network

**Congestion Detection:**

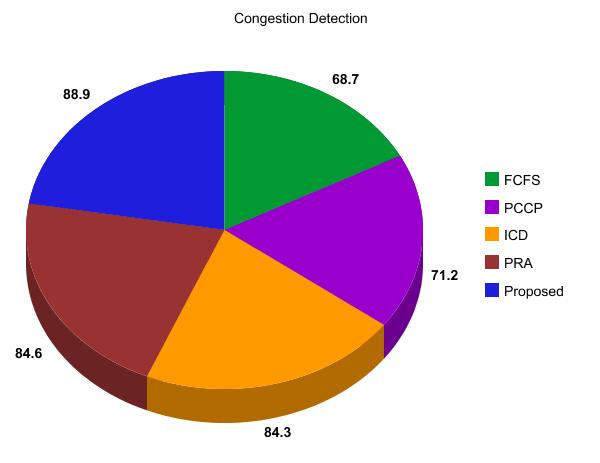
To detect congestion level we calculate packet service ratio i.e. PSR (%). It is the ratio of the average packet sending rate and the average packet scheduling rate at the transport layer in each node *i*, that is,

PSR(i) = (PSNDRi) / (PSCRi).

Here, PSNDRi also can be presented as (1/ ) where, is packet service time. Packet service time is the time difference between receiving time at the MAC layer and successfully forwarded to the next hop. can be calculated as:

= *w*i × + (1 - ) × last\_packet()

where last \_ packet( ) is the packet service time of thepacket that just has been forwarded and *ws* is a constantwhere 0 < *ws* < 1.

Figure 6: Congestion Detection

The proposed model has been implemented using Java Weka and shows the expected throughput-based fairness in terms of number of packets received at the destination. Moreover, it also shows that the queues are filled up by packets of different applications equivalent to their assigned priority. Also, it reduces remarkable overall packet loss than FCFS, PCCP, ICD and PRA due to its conservative nature which reserves slots in the queues for respective application types. Moreover, fairness is maintained even if the priority changes dynamically.

**CHAPTER-7**

**CONCLUSIONS**

Congestion control and fairness with dynamic priority for ad hoc networks are presented to address throughput based fairness among heterogeneous applications with different priorities. To achieve this, the proposed protocol dynamically updates the priority threshold variable to adapt required priority of the applications when needed. It also modifies the length of different queues to store packets according to their priority ratio. Performance evaluation through experiments results shows that the proposed work achieves good performance in dynamic scenarios for multiple numbers of applications. Moreover, implicit and explicit congestion control mechanism reduces packet loss which plays a pivotal role in throughput. As it updates the priority automatically, it can be used in the field of military control command operation it battlefield, rescue operation, areas including agriculture and emergency search operation successfully.

**References**

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. and Cayirci, E. (2002) ‘Wireless sensor networks: a survey’, *Computer* *Networks*, Vol. 38, No. 4, pp.393–422.

Bennett, J.C.R. and Zhang, H. (1996) ‘WF2Q: worst-case fair weighted fair queueing’, *Proc*. o*f IEEE INFOCOM’*96, *Fifteenth Annual Joint Conference of the IEEE Computer* *Societies*. N*etworking the Next Generation*, Vol. 1, pp.120–128.

Bharghavan, V., Lu, S. and Nandagopal, T. (1999) ‘Fair queuing in wireless networks: issues and applications’, *IEEE Personal Communications*, Vol. 6, No. 1, pp.44–53.

Demers, A., Keshav, S. and Shenker, S. (1989) ‘Analysis and simulation of a fair queueing algorithm’, *ACM* *SIGCOMM Computer Communication Review*, Vol. 19, No. 4, pp.1–12.

Ee, T.C. and Bajcsy, R. (2004) ‘Congestion control and fairness for many-to-one routing in sensor networks’, *Proc*. o*f the* 2*nd International Conference on Embedded Networked Sensor Systems*, Baltimore, Maryland, USA, pp.148–161.

Ganesh, P. and Chockalingam, S. (2012) ‘A collection of open research problems in rate based transport protocols for multihop ad hoc wireless network’, *Proc*. o*f International* *Conference on Advances in Engineering, Science and* *Management*, Nagapattinam, Tamil Nadu, India, pp.309–314.

Goyal, P., Vin, H.M. and Cheng, H. (1997) ‘Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks’, *IEEE/ACM Transactions on* *Networking*, Vol. 5, No. 5, pp.690–704.

Gungor, V.C., Akan, O.B. and Akyildiz, I.F. (2008) ‘A real-time and reliable transport (RT2) ‘Protocol for wireless sensor and actor networks’, *IEEE/ACM Transactions on Networking*, Vol. 16, No. 2, pp.359–370.

Hamid, M.A., Alam, M.M., Islam, M. and Hong, C.S. (2010a) ‘Enforcing fairness for data collection in wireless sensor networks’, *Proc*. o*f Annual Communication Networks and* *Services Research Conference*, Montreal, QC, Canada, pp.192 198.

Hamid, M.A., Alam, M.M., Islam, M., Hong, C.S. and Lee, S. (2010b) ‘Fair data collection in wireless sensor networks: analysis and protocol’, *Annals of Telecommunications*, Vol. 65, No. 7, pp.433–446.

Ramanathan, P. and Agrawal, P. (1998) ‘Adapting packet fair queueing algorithms to wireless networks’, *Proc*. o*f the 4th* *Annual ACM/IEEE International Conference of Mobile* *Computing and Networking, MobiCom ‘98*, Dallas, TX, USA, pp.1 9

Wang, C., Sohraby, K., Lawrence, V., Li, B. and Hu, Y. (2006) ‘Priority-based congestion control in wireless sensor networks’, *Proc*. o*f IEEE International Conference on Sensor* *Networks, Ubiquitous and Trustworthy Computing*, Washington, DC, USA, pp.22–31.

Yuan, X. and Duan, Z. (2009) ‘Fair round robin: a low complexity packet scheduler with proportional and worst-case fairness’, *IEEE Transactions on Computers*, Vol. 58, No. 3, pp.365–379.