

KADIR HAS UNIVERSITY  
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING



DYNAMIC MULTI THRESHOLD PRIORITY PACKET SCHEDULING  
ALGORITHMS FOR WIRELESS SENSOR NETWORKS

GRADUATE THESIS

SEZER UZUNGENÇ

May, 2015

Sezer UZUNGENÇ

M.S. Thesis

2015

DYNAMIC MULTI THRESHOLD PRIORITY PACKET SCHEDULING  
ALGORITHMS FOR WIRELESS SENSOR NETWORKS

by

Sezer Uzungenç

Bachelor's degree, Computer Engineering, Kadir Has University, 2012

Submitted to the Graduate School of  
Science and Engineering in partial fulfillment of the requirements for the degree of  
Computer Engineering  
Master of Science

Kadir Has University

2015

KADIR HAS UNIVERSITY GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

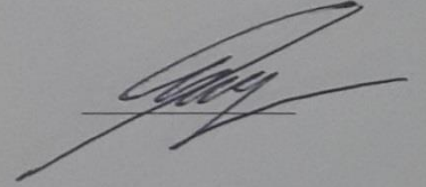
DYNAMIC MULTI THRESHOLD PRIORITY PACKET SCHEDULING ALGORITHMS  
FOR WIRELESS SENSOR NETWORKS

SEZER UZUNGENÇ

APPROVED BY:

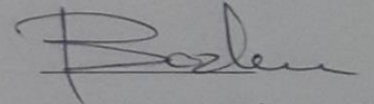
Asst. Prof. Tamer Dağ  
(Thesis Supervisor)

Kadir Has University



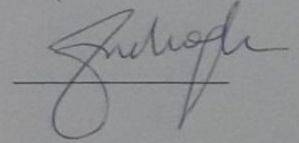
Assoc. Prof. Zeki Bozkuş

Kadir Has University



Asst. Prof. Tansal Güçlüoğlu

Yildiz Technical University



APPROVAL DATE: *May 13, 2015*

# DYNAMIC MULTI THRESHOLD PRIORITY PACKET SCHEDULING ALGORITHMS FOR WIRELESS SENSOR NETWORKS

## ABSTRACT

In Wireless Sensor Networks, it is needed to schedule different types of packets such as real time and non-real time packets. It is important to reduce sensors' energy consumptions and end-to-end data transmission delays. In this thesis, we propose new packet scheduling algorithms and integrate with Wireless Sensor Networks to improve energy consumptions and end-to-end data transmission delays. Our proposed Dynamic Multi Threshold Priority packet scheduling algorithms ensure a decrease in delay time and loss ratio for the lower priority level data with acceptable fairness towards higher priority level data. Threshold algorithms are compared with the commonly used scheduling algorithms such as First-Come-First-Serve (FCFS) and fixed priority non-preemptive. Simulation results illustrate that the Dynamic Multi Threshold Priority packet scheduling algorithms can provide a better QoS for low priority packets while keeping the QoS levels for high priority packets at similar levels.

# DYNAMIC MULTI THRESHOLD PRIORITY PACKET SCHEDULING ALGORITHMS FOR WIRELESS SENSOR NETWORKS

## ÖZET

Kablosuz sensör ağlarında farklı türlerde özellikle gerçek zamanlı ve gerçek olmayan zamanlı paket zamanlama gereklidir. Sensörlerin enerji kullanımlarını ve iletim gecikmelerini azaltmak önemlidir. Tezimde yeni paket zamanlama algoritmalarını geliştirerek bunu kablosuz sensör ağlarına entegre etmeye çalışarak enerji kullanımını ve iletim gecikmelerini geliştirerek daha verimli yapıyorum. Tasarladığım dinamik çoklu eşik ve öncelikli paket zamanlama algoritmaları, düşük öncelikli veriler için gecikme zamanını ve veri kaybını azaltarak bunu yüksek öncelikli verilere adil bir şekilde davranarak yapıyor. Eşik algoritmaları günümüzde en çok kullanılan paket zamanlama algoritmalarıyla kıyaslanıyor. Bunlar ilk gelen ilk servis edilir algoritması ile öncelikli paket zamanlama algoritmasıdır. Simülasyon sonuçları gösteriyor ki dinamik çoklu eşik ve öncelikli paket zamanlama algoritmaları düşük öncelikli verilerin servis kalitesini arttırıyor ve bunu yüksek öncelikli verilerin servis kalitesini koruyarak yapıyor.

## **ACKNOWLEDGEMENTS**

I would like to thank Asst. Prof. Tamer Dağ for giving me a chance to work with him. He helped me in building my academic skills with his extensive knowledge and outstanding research profile. He let me have a comfortable working environment with his patience, tolerance and understanding. I learned a lot from him and I would like to owe my deepest gratitude to this wonderful supervisor.

Finally, I thank my family and all friends who supported me both during my studies and in writing my thesis.

*To my family...*



## TABLE OF CONTENTS

ABSTRACT .....	3
ÖZET .....	4
ACKNOWLEDGEMENTS .....	5
LIST OF TABLES .....	9
LIST OF FIGURES .....	10
LIST OF ABBREVIATIONS .....	12
1 INTRODUCTION.....	13
2 DISCRETE EVENT SYSTEM SIMULATION .....	15
2.1 What is Simulation?.....	15
2.2 Steps in a Simulation Study.....	15
2.3 Computer Simulation.....	18
2.4 Discrete Event System (DES) Simulation .....	18
2.4.1 Concepts in a Discrete Event Systems .....	19
2.4.2 The Event-Scheduling .....	21
2.5 Queuing Simulation.....	23
2.5.1 Waiting Line Models.....	24
3 SCHEDULING ALGORITHMS .....	26
3.1 What is Scheduling?.....	26
3.1.1 Main Concern of Scheduling.....	26
3.2 Types of Scheduling Algorithms .....	27
3.2.1 First-Come-First-Serve (FCFS) Scheduling.....	27
3.2.2 Fixed Priority Non-Preemptive Scheduling .....	27
3.2.3 Shortest Remaining Time Scheduling.....	28
3.2.4 Round Robin Scheduling .....	28

3.2.5	Multilevel Queue Scheduling.....	28
3.2.6	Fixed Priority Preemptive Threshold Scheduling .....	29
4	MULTI-THRESHOLD SCHEDULING.....	30
4.1	Multi Threshold Priority Packet Scheduling .....	30
4.2	Dynamic Multi Threshold Priority Packet Scheduling.....	37
4.3	Dynamic Multi Threshold Priority Packet with Urgency Packet Scheduling .....	41
4.4	Implementation of the Algorithms .....	46
4.4.1	Main Program.....	53
4.4.2	Initialization .....	54
4.4.3	Arrival Method .....	55
4.4.4	Departure Method .....	57
4.4.5	Report Generation .....	57
5	ANALYSIS OF MULTI-THRESHOLD SCHEDULING ALGORITHMS .....	59
5.1	First Phase – Total Loss Ratio Analysis.....	60
5.2	Second Phase – Loss Ratio Analysis for each Priority Level.....	62
5.2.1	Priority-1 Level Loss Ratio Analysis .....	68
5.2.2	Priority-2 Level Loss Ratio Analysis .....	70
5.2.3	Priority-3 Level Loss Ratio Analysis .....	72
5.3	Third Phase – Total Delay Time Analysis.....	74
5.4	Fourth Phase – Delay Time Ratio Per Each Priority Level.....	76
5.4.1	Priority-1 Level’s Delay Time Ratio Analysis.....	77
5.4.2	Priority-2 Level’s Delay Time Ratio Analysis.....	79
5.4.3	Priority-3 Level’s Delay Time Ratio Analysis.....	81
5.5	Total Evaluation .....	83
6	CONCLUSIONS AND FUTURE WORK .....	85
7	REFERENCES.....	88
8	CIRRICULUM VITAE.....	90

## LIST OF TABLES

Table 4.1 - Urgent Status.....	42
Table 4.2 - Definition of variables. ....	49
Table 4.3 - Definition of methods .....	51
Table 4.4 - Definition of functions.....	51

## LIST OF FIGURES

Figure 2.1- Steps in a simulation study. ....	17
Figure 2.2 - Flowchart of time-advance algorithm. ....	23
Figure 2.3 - Single-server queuing system. ....	24
Figure 2.4 – Overall structure of system. ....	25
Figure 4.1 – Basic threshold structure at the waiting queue. ....	31
Figure 4.2 - Multi-threshold system for multi-priority events at the waiting queue. ....	32
Figure 4.3 - Arrival event of Multi Threshold Priority Scheduling algorithm. ....	33
Figure 4.4 – Addition of packet to in front of threshold. ....	34
Figure 4.5 – Addition of packet to instead of lower priority. ....	35
Figure 4.6 – Addition of packet to behind of threshold. ....	35
Figure 4.7 – Dropping of lower priority arrival packet. ....	36
Figure 4.8 - Arrival event of Dynamic Multi Threshold Priority Scheduling algorithm. ....	39
Figure 4.9 - Arrival event of Dynamic Multi Threshold Priority with Urgency Scheduling... ..	43
Figure 4.10 - Sliding with arrival of the highest priority packet enters to system. ....	44
Figure 4.11 - Event scheduling program’s structure. ....	48
Figure 4.12 - Structure of simulation. ....	52
Figure 4.13 - Event Class ....	55
Figure 4.14 - Arrival event of Priority Queue scheduling algorithm. ....	56
Figure 4.15 - Departure event of Priority Queue scheduling algorithm. ....	57
Figure 5.1 - The total loss ratio - capacity graph for all algorithms, when throughput value is 0.95. ....	61
Figure 5.2 - The total loss ratio - capacity graph for all algorithms, when throughput value is 0.99. ....	62
Figure 5.3 - Loss ratio - Priority graph for all algorithms, when throughput value is 0.95 and capacity is 50. ....	63
Figure 5.4 – Loss ratio - Priority graph for all algorithms, when throughput value is 0.95 and capacity is 100. ....	64
Figure 5.5 - Loss ratio - Priority graph for all algorithms, when throughput value is 0.99 and capacity is 50. ....	65

Figure 5.6 – Loss ratio - Priority graph for all algorithms, when throughput value is 0.99 and capacity is 100.....	67
Figure 5.7 - Loss Ratio (Priority 1) - Capacity graph, when throughput value is 0.95.....	68
Figure 5.8 - Loss Ratio (Priority 1) - Capacity graph, when throughput value is 0.99.....	69
Figure 5.9 - Loss Ratio (Priority 2) - Capacity graph, when throughput value is 0.95.....	70
Figure 5.10 - Loss Ratio (Priority 2) - Capacity graph, when throughput value is 0.99.....	71
Figure 5.11 - Loss Ratio (Priority 3) - Capacity graph, when throughput value is 0.95.....	72
Figure 5.12 - Loss Ratio (Priority 3) - Capacity graph, when throughput value is 0.99.....	73
Figure 5.13 - Total Delay Time Analysis for all algorithms with different options .....	74
Figure 5.14 - Average Delay Time (Priority 1) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms. .....	77
Figure 5.15 - Average Delay Time (Priority 1) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms. .....	78
Figure 5.16 - Average Delay Time (Priority 2) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms. .....	79
Figure 5.17 - Average Delay Time (Priority 2) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms. .....	80
Figure 5.18 - Average Delay Time (Priority 3) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms. .....	81
Figure 5.19 - Average Delay Time (Priority 3) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms. .....	82

## **LIST OF ABBREVIATIONS**

FEL – Future Event List

MTPS - Multi Thresholds Fixed Non Pre-Emptive Priority Scheduling

DMTPS - Dynamic Multi Thresholds Fixed Non Pre-Emptive Priority Scheduling

DMTPUS – Dynamic Multi Thresholds Fixed Non Pre-Emptive Priority with Urgent Scheduling

MLQ – Multi-level Queue

DQ – Drop Queue

FCFS - First-Come-First-Served

FIFO – First-In-First-Out Scheduling Algorithm

P - Process

QoS – Quality of Service

WQ – Waiting Queue

# 1 INTRODUCTION

A Wireless Sensor Network (WSN) consists of tiny sensor nodes which are low-cost and low-energy and with sensing, data processing and communicating components. A WSN can be realized with many sensor nodes that are deployed closely. The feature of sensor networks is the cooperative effort of the sensor nodes. Sensor nodes are fitted with an onboard processor. Instead of sending the raw data to the nodes responsible for the fusion, they use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data [1].

WSNs are used in various areas such as military, health and home. For military applications, the rapid deployment, self- organization and fault tolerance characteristics of WSNs make them a very promising sensing technique for military command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting systems. In health, they are used for monitoring patients and assisting disabled patients with deployment of sensor nodes. Managing inventory, monitoring product quality, and monitoring disaster areas are some other applications where WSNs are used [1].

In WSNs, sensor nodes can be mobile and heterogeneous. They are scalable to large scale of deployment. Design of a WSN is cross-layer and it is becoming an important study area for wireless communications. Cross layer design is used for improving the transmission performance which is energy efficiency, data rate and Quality of Service (QoS).

WSNs may be deployed in various environments and energy determines the lifetime of a WSN. Communication in sensor nodes is a key component and protocols and algorithms must satisfy the issues such as increasing of lifespan, self-configuration, robustness and fault tolerance. WSNs have a problem with energy consumption. Energy is the most important resource of WSN nodes [2], [3].

Scheduling is the strategy by which the system decides which task should be executed at any given time. Generally, it is used for load balance and system resources to be served effectively and with desired quality [4]. Scheduling algorithms are classified as preemptive and non-preemptive. In preemptive scheduling, arrival of high priority data suspends current

lower priority data. In non-preemptive scheduling, the current lower priority data is processed until it is completed. Preemptive schedulers provide better performance for system utilization. Non preemptive schedulers encounter the starvation problem [5]. Non preemptive schedulers may give better performance and robustness on reduced systems [6].

The major scheduling algorithms for WSNs are FCFS and fixed priority non-preemptive scheduling. In FCFS algorithm, each priority level of data's loss ratio and delay time results are assumed to be the same since priority is not criteria in this algorithm [7]. In fixed priority non-preemptive scheduling algorithm, lower priority level data can get lost and delayed excessively and the higher priority level data are processed more quickly and there is a definite unfairness between priority levels [8].

In this thesis, we develop new scheduling algorithms for WSNs which can provide a better QoS and resolve the disadvantages of the current algorithms. The proposed Dynamic Multi Threshold Priority packet scheduling algorithms try to maximize the level of QoS provided to different priority packets by reducing the loss ratio and waiting times. Also, the proposed scheduling algorithms are non-preemptive due to structure of reduced system load of WSNs.

The rest of the paper is organized as follows: Chapter 2 introduces with Discrete Event System Simulation, Chapter 3 describes Scheduling and types of scheduling algorithms are specified, our proposed Dynamic Multi Threshold Priority Packet Scheduling algorithms and implementation of algorithms are explained in Chapter 4. Simulation results of the algorithms are shown in Chapter 5 and finally Chapter 6 concludes this paper and suggests for future researches on that thesis.



## **2 DISCRETE EVENT SYSTEM SIMULATION**

In this chapter, discrete event system simulation is explained with all features. Concepts in Discrete Event System, The-Event Scheduling and Queuing Simulation are clearly explained for this thesis work.

### **2.1 What is Simulation?**

Simulation is an integration of real time events to system with operations. First, it needs to develop a model. This model's main characteristics or behaviors are applied to chosen physical system or process. Model represents itself, but simulation represents the operation of system in over time.

It is used for very different situations. For example, it is used technology for performance optimization. Video games and education are the examples of that. Computer experiments frequently are used and tested in simulation model. Simulation is also used for gaining function of human systems scientific model. Also, alternative conditions and real effect of courses of action are used. Simulation can be used when in real systems are not used [19].

### **2.2 Steps in a Simulation Study**

These are the phase of the simulation study;

1. Problem formulation: Determine the problem clearly.
2. Setting of objectives and overall project plan: How to approach our problem?
3. Model conceptualization: Set up a reasonable model.
4. Data collection: Gather essential data and simulate.(arrival rate, process rate)
5. Model translation: Turn model into programmable language.
6. Verification: Control the program and approve if it works correctly.
7. Validation: Control the system that works correctly and have realistic outputs.

8. Experimental design: Query the system that how is it run with which type of input variation. System time is important.
9. Production runs and analysis: Gather the output and analyze.
10. Repetition: Keep on trying if it is necessary.
11. Document and report: Documenting it and report the result.
12. Implementation: If previous steps perform successfully, then implement it.

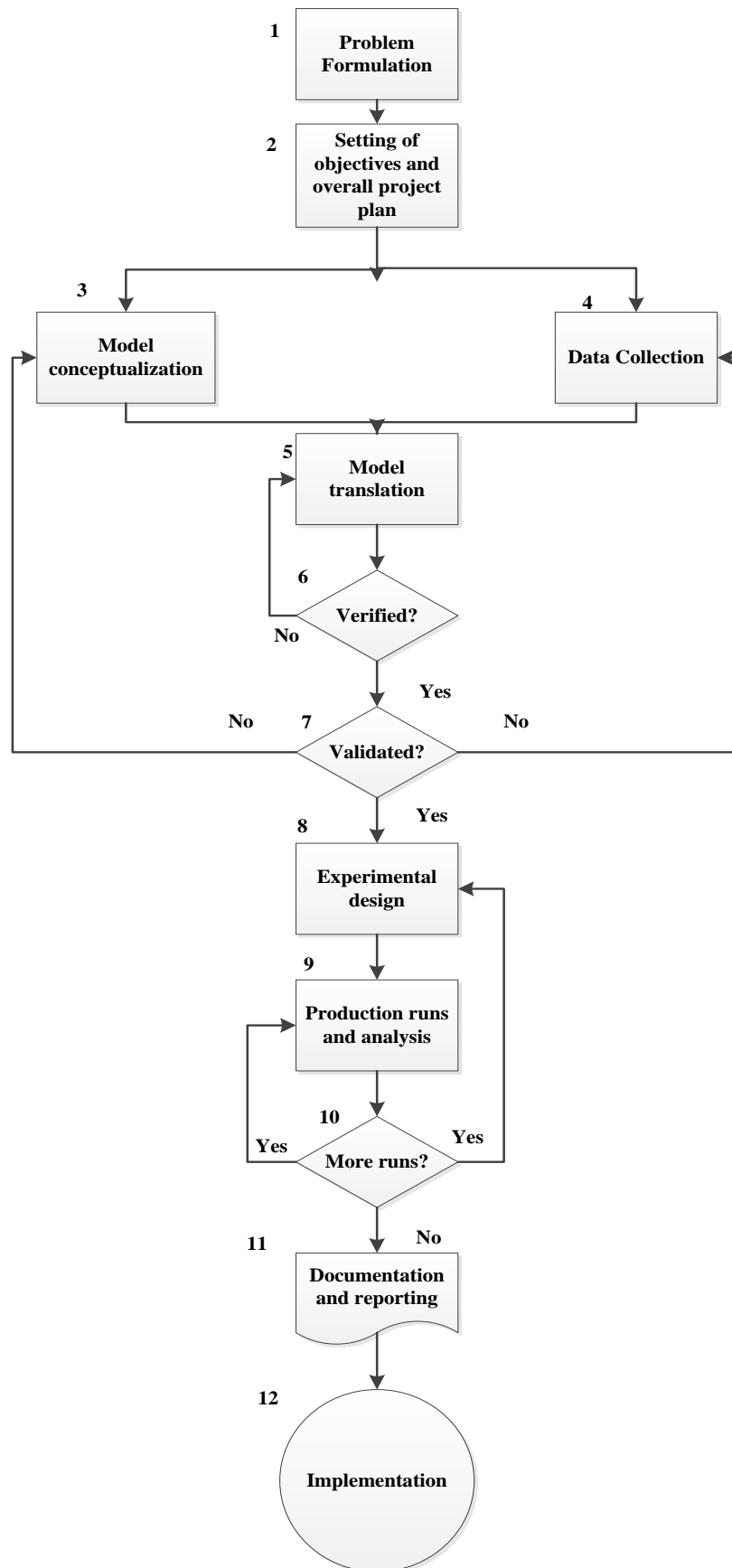


Figure 2.1- Steps in a simulation study.

### **2.3 Computer Simulation**

Computer simulation is an experience in modelling real life situations how operate and response in computer. Predictions can be done in simulation and it is experienced how system reacts to changing variables. It is a tool for researching of systems behavior.

Computer simulation turns into very beneficial situation for many natural systems to be experienced (physics, chemistry and biology). Using computer simulation, network traffic simulation which has high utility can be experienced and used. Initial parameters are important for model behavior and depend on it. It may vary for each simulation.

Generally, mathematic modelling is a formal modelling of system. Initial condition and parameters try to find analytical solutions for prediction of sequence system behavior. Usually for modelling systems, it is not possible closed form of analytical solutions.

### **2.4 Discrete Event System (DES) Simulation**

Discrete-Event System Simulation is a simulation that models the systems which consist of discrete sequence of events in time. Each event is evaluated by arrival times in system and it provides to change in system states. There is not any change in system between consecutive events and simulation passes an event to other new event. Arrival event calls new arrival event for system continuity and system goes on.

System is changed according to inside of dynamics and it is tracked. Using activity based simulation is different from event based simulation. Time is separated into small pieces and system state is updated according to inside of small pieces activities. It is not need to simulate for each time pieces in discrete event simulation because it runs faster than responded continuous simulation.

Other alternative for event based simulation is process based simulation. In this simulation, each activity in system responds each process. It is simulated by thread. In discrete events, generated by thread processes may cause other thread process that change (sleep, wake or update the state).

The most commonly used discrete event simulation method is three-phased approach. First phase is passes into chronological event. In second phase, all events are run in unconditionally time. In the last phase, all events are run in conditionally at time. This three-phased simulation is a most effective system in event simulation for computer resources. Usually, it is used in advertising software packages and according to user's point of view it changes and principal places is hidden from user, specifically.

### 2.4.1 Concepts in a Discrete Event Systems

Concepts in Discrete Event Systems are system, model, system state, entity, attributes, set, event, activity and delay.

- *System*: Entities are interacted with themselves for one or more goals in times.
- *Model*: It is an abstract of system representation. Usually, it locates logical and/or mathematical relations. This is a describing of a system in terms of state, entities and their attributes sets, events, delays and activities.
- *System State*: It is a collection that collects all information about variable and defines in any time.
- *Entity*: It is a model that any object or component presents openly in system (server, customer, machine).
- *Attributes*: It is a property of given entities (priority of a packet, the routing of a job through a job shop).
- *Set*: It is an order of permanent or temporarily associated logical order (for example, all customers currently waiting in line ordered by first come first served).
- *Event*: An instant event changes system status.
- *Activity*: Starting time and its length are known that duration of time of specified length (inter arrival time, service time).
- *Delay*: Completed time and its length are not known that duration of time of unspecified length.

Different terminology may be operated for different packages. Queue, chain or set is expressed with lists. Rules are specific for ordering entities on a list. The main rules of order

are first come first served and ranked by priority. Event time is important for ordering event time, event notice records and Future Event List [10].

The main operations on a list are;

- Removing a record from the top of the list
- Removing a record any location on the list
- Adding a record to the top or the bottom of the list
- Adding a record at an arbitrary position in the list.

Inter arrival time, service time or other processing time is represented with an activity. It is specified with some techniques.

- Deterministic - It specifies exact time of activity.
- Statistical - It specifies random draw from set with equal probability.
- Function that depends on entity or variable -It specifies executing function that may be statistical distribution.

Tracking and predicting activity's finishing time begin with starting of simulated activity duration's time. Creating of event notice event time and activity's completion time is equaled.

System conditions are very important for delay durations. Delay durations are assigned with conditions not with modeler. On Future Event List, event notice's place is managed with primary event and provided with completing of activity. Secondary event is the completion of a delay.

All functions of time, system state and the activities changes with time. Time which means simulation time represented with Clock.

Producing sequence of systems is necessary for continuity of discrete event systems. System time at  $t$  includes system state, progressed activities on Future Event List and current value of statistics that calculates summary statistic at end of the simulation.

### 2.4.2 The Event-Scheduling

Event Scheduling is used for ordering all events by time on the Future Event List. All event notices which are scheduled to occur at a future time are located in Future Event List. Activity starts at an instant time and durations are calculated or defined with distribution. It is assigned with the last activity event time and it is placed on the Future Event List. Most of the future events are not scheduled and just occurred with random arrivals in real life. End of activities are necessary for representing random events. It is provided with statistical distributions. Clocks value is  $t$  which means time. It represents current value time of simulation. Imminent event is  $t_1$ . If it is called, then next event should be called. After that, Clock value is updated and new values of simulation time is  $t_1$  instead of  $t$ . In Future Event List, imminent event is removed. Then, next event is executed. New system images are created with the executing of imminent event. New events may be generated at time  $t_1$ . They are scheduling with creating of event notice and placed appropriately, if new events do not exist. CLOCK gets new value which is the new imminent event's time and event is executed, after of updating system image for  $t_1$ . This process are proceeded to end of simulation.

While simulation runs over, Future Event List's length and content continuously change. If efficient management is not used, simulation gets in a complex state. List processing is necessary for management. The most used list processing is the removal of imminent event operation. Then, new packet is added to list and some of the events are removed. Generally, placement in the future event list is specified with imminent event in the list. It is placed at the top of the list and definitely should be removed first. Search is required for removing old event or adding a new event. It depends on the logical organization. All the sets must be placed in an order and list processing techniques are required for removing entities.

Imminent event is removed from the Future Event List and its time advances to simulation time value. The new arrival event is generated and added to Future Event List. The second arrival event is generated. Inter arrival time of event, let's say  $a^*$  is generated and added to current time. The result of future event time is equaled to  $t + a^*$  and it is added to event list appropriate place.

At  $CLOCK = t$ , if the next event's departure are demanded to be scheduled, then it is done with service time  $s^*$  and it is scheduled at  $t + s^*$  to occur [10].

End of simulation is done with stopping event. The stopping event is created with criteria of user. Let's say, if the system runs for 10000 event, 10000.event is assumed as stopping event.

Event-scheduling / time-advance algorithm:

Step 1 - Remove the event notice for the imminent event (event 3, time  $t_1$ ) from FEL

Step 2 - Advance  $CLOCK$  to imminent event time (i.e. advance clock from  $t$  to  $t_1$ )

Step 3- Execute imminent event: update system state, change entity attributes and set membership as needed.

Step 4 - Generate future events (if necessary) and place their event notices on FEL, ranked by event time. (Example: Event 4 to occur at time  $t^*$ , where  $t_2 < t^* < t_3$  )

Step 5 - Update cumulative statistics and counters [10].



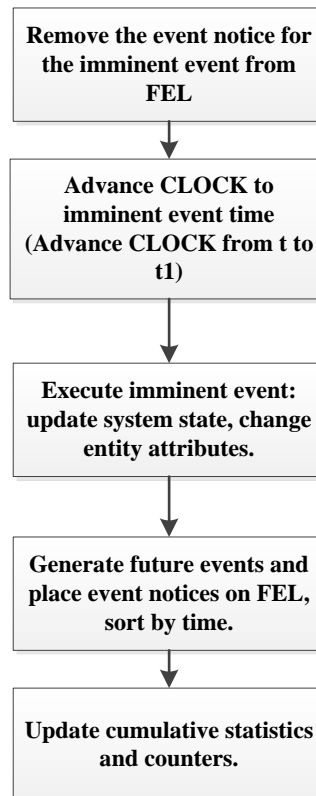


Figure 2.2 - Flowchart of time-advance algorithm.

## 2.5 Queuing Simulation

In this chapter, it is explained that how the simulation is queuing. Waiting line models and events are clearly explained for figuring out how to queuing simulation is used in thesis work.

Dynamic and event-based models are;

- A single server queue
- A two server queue

Waiting line models in a single server queue is expressed at Chapter 2.5.1 for using in this thesis work.

### 2.5.1 Waiting Line Models

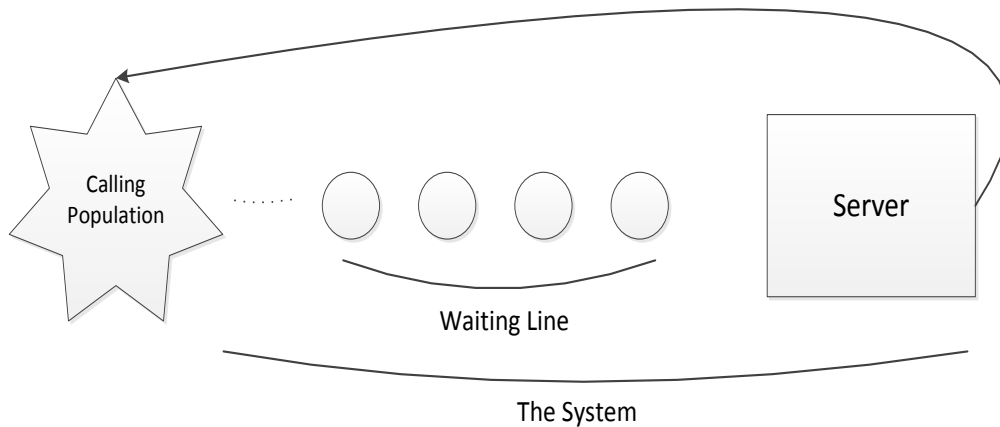


Figure 2.3 - Single-server queuing system.

- In simulation of queuing system, it is infinite to call population.
- Arrivals are determined with statistical distribution and it occurs one at a time.
- Service times are determined with statistical distribution.
- The system capacity is limited.
- Callers are served in some ways; First-Come-First-Served, priority queuing etc.

Effective system should consists of arrival rate < service rate. If it is not, system is explosive or unstable. There are two events:

- Arrival Event: Event that enters into a system.
- Departure Event: Event that completes its service.

Arrival events are placed to waiting queue, if it is not full. Events are ordered by such as First-In-First-Out, Priority queuing, Shortest Remaining Time etc. Queue capacity is important and drop operation is done if the capacity of Waiting Queue is full. Server is a section which utilizes events and finishes its tasks. When server is empty, an event which places in Waiting Queue is called to server and event's scheduling of departure is begun. Server is indicated with flags 0 and 1. If server is full, 1 is assigned to Process ( $P$ ). If server is empty and there is not any event in Waiting Queue, then flag value of  $P$  is assigned to 0.

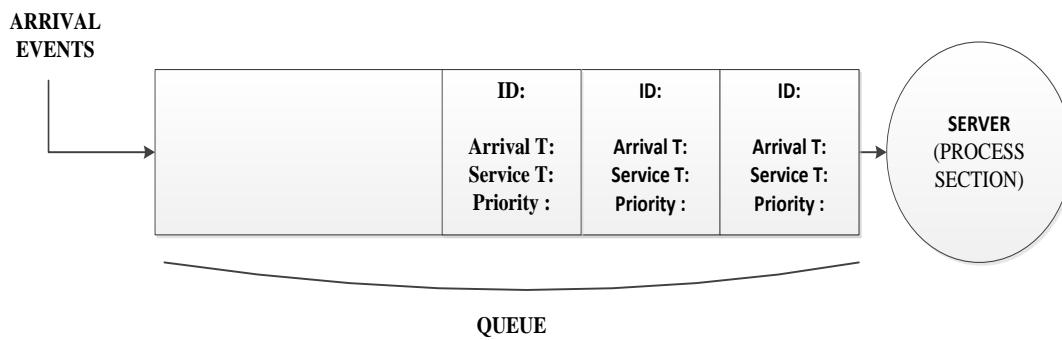


Figure 2.4 – Overall structure of system.

### 3 SCHEDULING ALGORITHMS

In this chapter, meaning of scheduling is described and types of scheduling algorithms are introduced.

#### 3.1 What is Scheduling?

Scheduling is a method for access system resources with using dataflow, threads and processes. Generally, it is used for load balance and system resources to be served effectively and desired quality. Most modern systems are multi-tasking (multi process at same time) and multiplexing (real time data communication with physical channel) [15]. Effectively using of scheduling is a key point with increasing of requirement.

##### 3.1.1 Main Concern of Scheduling

Generally, scheduler is interested with these parameters.

*Throughput* - It is a ratio of completed process in specific time.

*Turnaround time* – It is a total of time between beginning process in submission time and its finish.

*Response time* - It is a total of time between request of a submission time and produce of first response time.

*Fairness* - Its purpose is to give each process to equal CPU Time. The same importance and work force for each process.

*Waiting Time* - It is a time of process which remains in queue.

A scheduler implements suitable compromise when the goals conflict. The goals conflict rarely. Preference is given to anyone by user's needs and goals.

For an automatic control embedded systems, schedulers should ensure that can meet with its deadlines in real time environment. It is very important for a system that it carries on with a stable way. Scheduled tasks can be divided into remote devices with network and it can be managed in administrative back end.

## 3.2 Types of Scheduling Algorithms

Some of the scheduling algorithms are First-Come-First-Serve, Fixed Priority Non Pre-Emptive, Shortest Remaining Time, Round-Robin, Multilevel Queue and Fixed Priority Preemptive Threshold Scheduling.

### 3.2.1 First-Come-First-Serve (FCFS) Scheduling

It is known as First-In-First-Out (FIFO). It is the easiest scheduling algorithm. In First-In-First-Out scheduling algorithm, packets are put to queue easily and packets wait ready in queue according to order of arrival time.

- There is not any placement by packets priority order
- Queuing is based.

FCFS algorithm has a disadvantage on very high loss ratio and long average delay time for higher priority real-time data. Data packets are ordered by arrival times and priority is not important criterion for ordering [7]. In this algorithm, large delay time for higher priority data can be experienced. Also, real time packets might not receive timely service and cannot supply delay time guarantees.

### 3.2.2 Fixed Priority Non-Preemptive Scheduling

In fixed priority non-preemptive scheduling algorithm, each arriving data packet has a defined priority level. The scheduler arranges the packets which are ready and ordered by their priority levels in queue [8]. Lower priority packets can be dropped excessively when higher priority packets arrive at the queue. Fixed priority non-preemptive Scheduling algorithm has an advantage on higher priority real-time data packets in comparison to the similar packets in the FCFS algorithm. Loss ratio and average waiting time of higher priority data results are better than FCFS algorithm, but lower priority data's results encounters with problem. Lower priority data's are processed last and they can be lost with the arrival of higher priority data so that lower priority data can be lost excessively. Starvation of lower priority data occurs at the system and it is unfairness. Also, transmission of a large data packet

in non-preemptive priority scheduling algorithm incur starvation of a high priority real-time data packets [9].

### **3.2.3 Shortest Remaining Time Scheduling**

In shortest remaining time scheduling algorithm, queue is set by scheduler according to packets who have a least time. Calculation of estimating time and advanced knowledge is necessary for completion of process [16].

### **3.2.4 Round Robin Scheduling**

In round – robin scheduling algorithm, fixed time is assigned to each packet in circular order by scheduler.

- It is lead to extensive overhead special in less system time.
- In First-Come-First-Served scheduling algorithm, shorter jobs can be processed quickly according to Round-Robin scheduling algorithm.
- It is gained well average time on response times.
- Number of packets is important for waiting time and it is dependent [17].

### **3.2.5 Multilevel Queue Scheduling**

When packets are easily divided and entered into groups, multi-level queue scheduling algorithm is used. If there are two type packets which are interactive packets and batch packets, there are different response time requirement and it must be scheduled by different scheduling. It is very important for this scenario because it shares memory problems.

Multilevel queue (MLQ) scheduling is very complex scheduling technique which makes use of basic CPU scheduling techniques available. MLQ scheduling processes are assigned to a fixed queue. In literature we will find by how many ways we can divide our queues, the

criteria and terminology that are used till now to schedule CPU resource using the MLQ technique [18].

### **3.2.6 Fixed Priority Preemptive Threshold Scheduling**

In fixed priority pre-emptive threshold scheduling, each packets has proper priority and each priority has a threshold not as a capacity of the system. Scheduler arranges packets readily. Lower priority packets may have lower threshold values and it can be interrupted.

Preemption-threshold allows a thread to only disable preemption of threads up to a specified threshold priority. Threads having priorities higher than the threshold are still allowed to preempt. By utilizing preemption-threshold, high-priority threads can continue to respond - in real-time - to the external world and not be blocked by unrelated critical section processing of lower-priority threads [13], [14].

## 4 MULTI-THRESHOLD SCHEDULING

In this chapter, algorithms of thesis work are presented. In this thesis work, multi threshold priority packet scheduling algorithms are developed. Three multi threshold priority packet scheduling algorithms are developed. One of the developed scheduling algorithms has non-dynamic threshold structure and others have dynamic threshold structure. In this thesis work, threshold definition is completely different from Fixed Priority Preemptive Threshold Scheduling (Chapter 3.2.6) which is currently used.

The developed algorithms are Multi Threshold Priority Scheduling (MTPS), Dynamic Multi Threshold Priority Scheduling (DMTPS) and Dynamic Multi Threshold Priority with Urgency Packet Scheduling (DMTPUS).

### 4.1 Multi Threshold Priority Packet Scheduling

The first algorithm of multi threshold priority scheduling is MTPS. Packets are got into waiting queue by their priorities and each priority level has own threshold. The basic threshold is shown at Figure 4.1.

The main aim of the Multi Threshold Fixed Priority Scheduling is to provide order of each priority level with its threshold. The most important operation of the MTPS is to locate lower priority packets. If front of appropriate threshold is full and lower priority packet is found at the threshold, then it is changed with the higher priority.



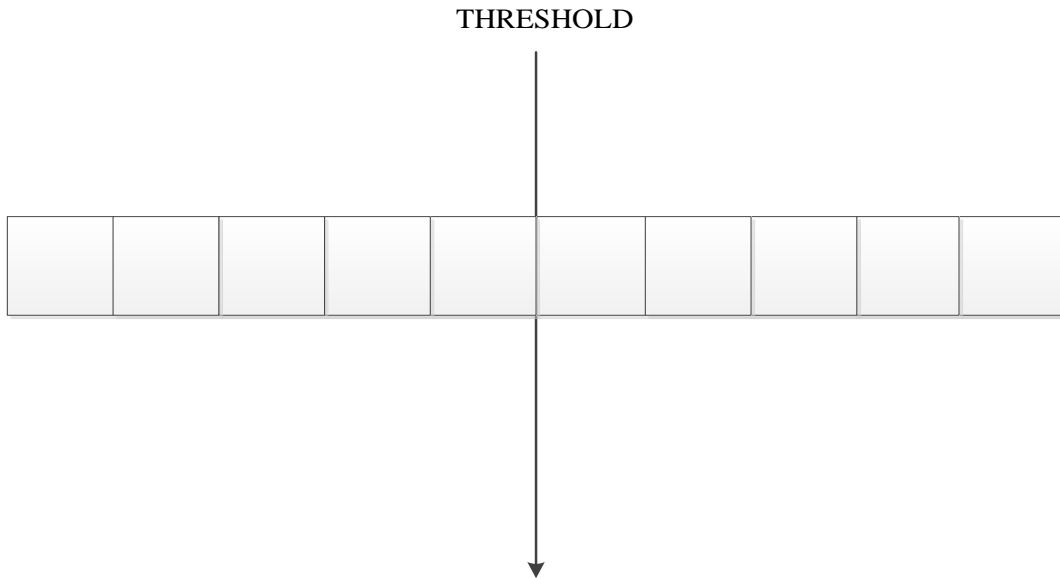


Figure 4.1 – Basic threshold structure at the waiting queue.

The threshold values for each priority level are specified with different values. Priority levels are described as the high, medium and the low level. Each packet has proper priority and assignment of packet's priority operation is similar with fixed priority non-preemptive scheduling algorithm.

Priority – 1 (The high priority level):  $\%k$  of capacity in the waiting queue.

Priority – 2 (The medium priority level):  $\%m$  of capacity in the waiting queue.

Priority – 3 (The low priority level):  $\%n$  of capacity in the waiting queue.

Threshold values are specified as  $n > m > k$ . These values can be changed depending on the traffic situation in the network.

The high priority level data's own threshold value is described as  $k$  and it should be lower than lower priority level data's own threshold values which are  $m$  and  $n$ . It should have a lower value because fairness of higher priority level data towards the lower priority level data.

Each priority level has a unique threshold level. Higher priority levels' assigned threshold level can change with lower priority level's assigned threshold level if higher

priority data could not get into assigned threshold level. Figure 4.2 shows the representation of initial values of threshold values.

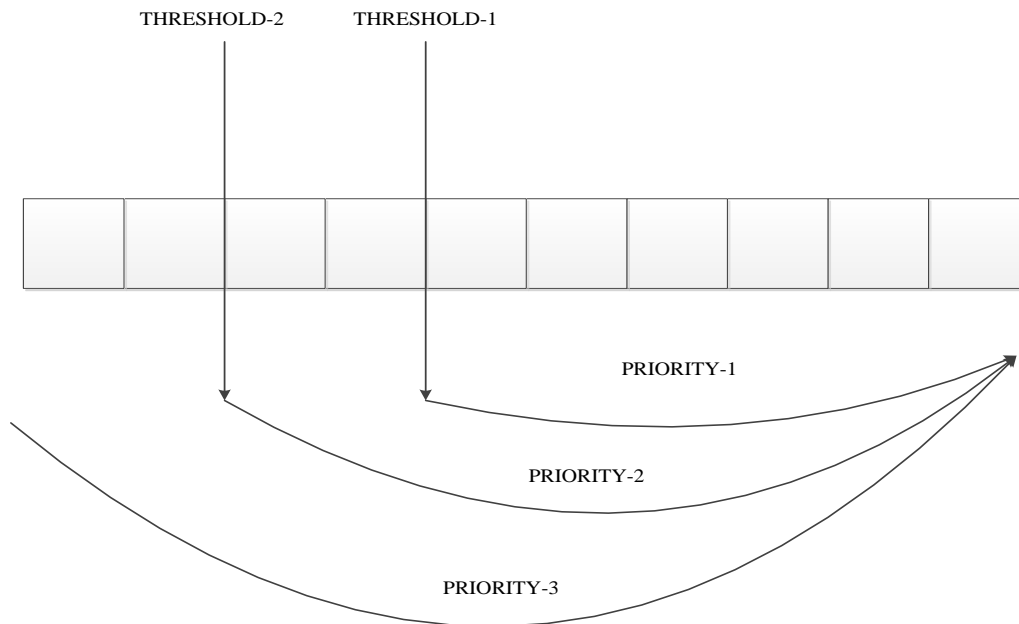


Figure 4.2 - Multi-threshold system for multi-priority events at the waiting queue.

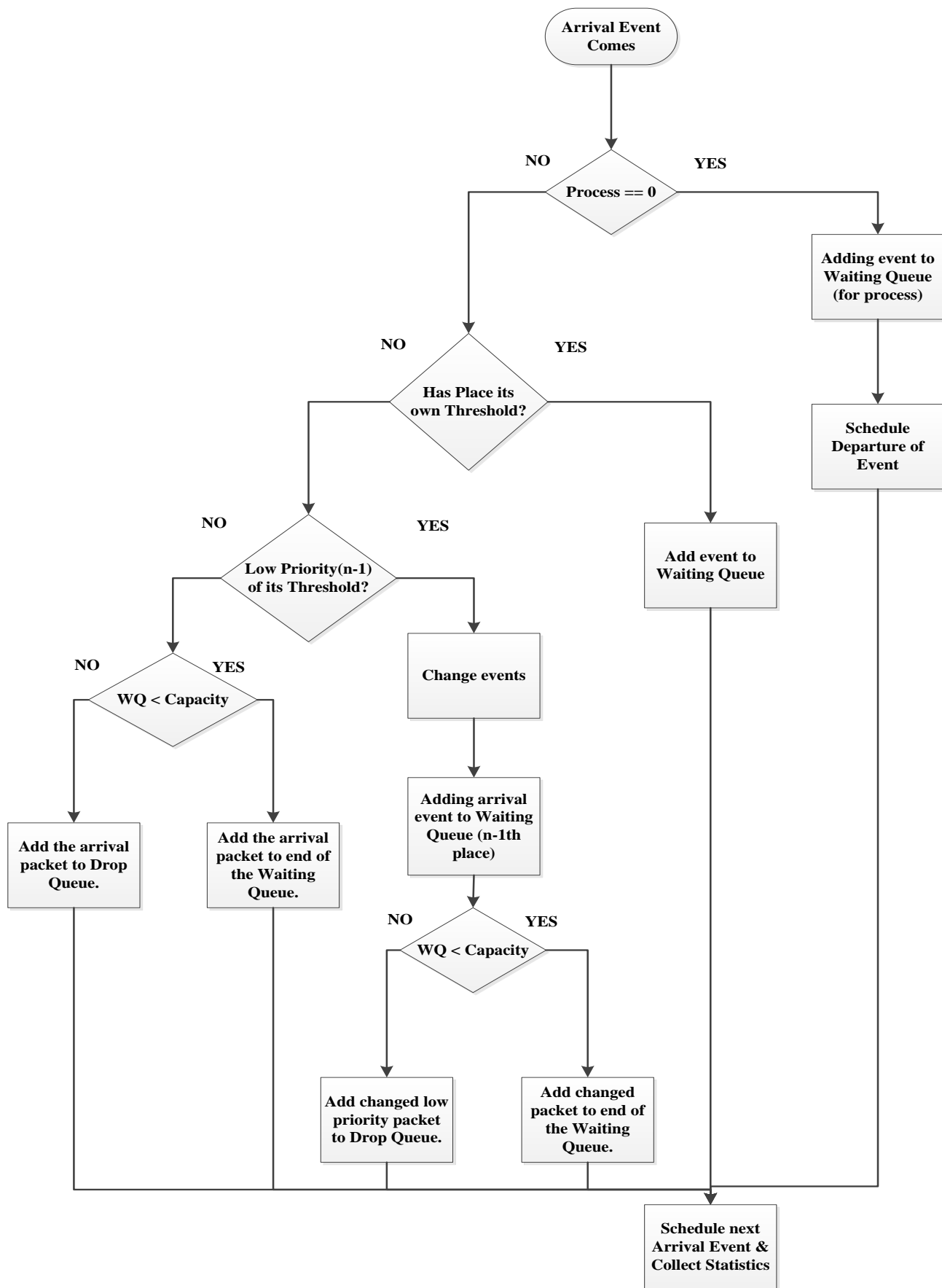


Figure 4.3 - Arrival event of Multi Threshold Priority Scheduling algorithm.

Server and waiting queue is full or empty it is controlled. Server is expressed with P and waiting queue is expressed with WQ. Arrival packets go into the system and P and WQ value assigns its next move.

Algorithm of arrival event follows as;

When the Server is empty:

The arrival packet goes into the Server directly without going into the Waiting Queue. Departure of the packet is scheduled and the arrival packet's departure process is started. The new arrival packet is called for Discrete Event System's continuity.

When the Server is full:

Each threshold capacities are defined. The arrival packet's priority and its own threshold value are analyzed. The packet's appropriate threshold value is assigned by its priority.

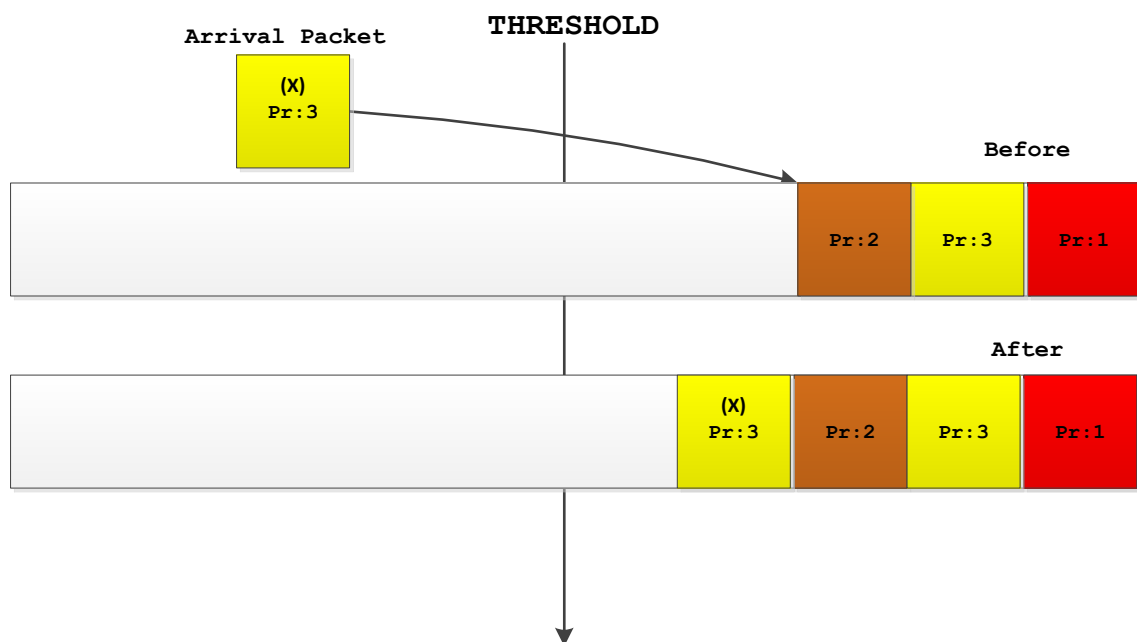


Figure 4.4 – Addition of packet to in front of threshold.

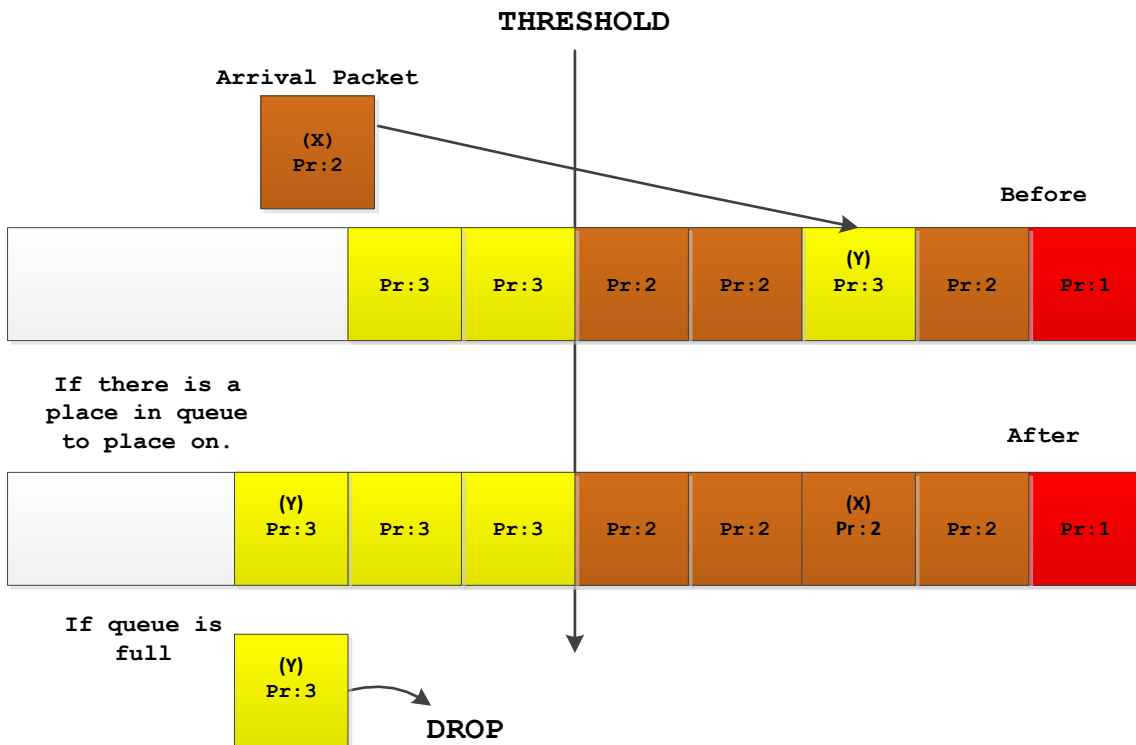


Figure 4.5 – Addition of packet to instead of lower priority.

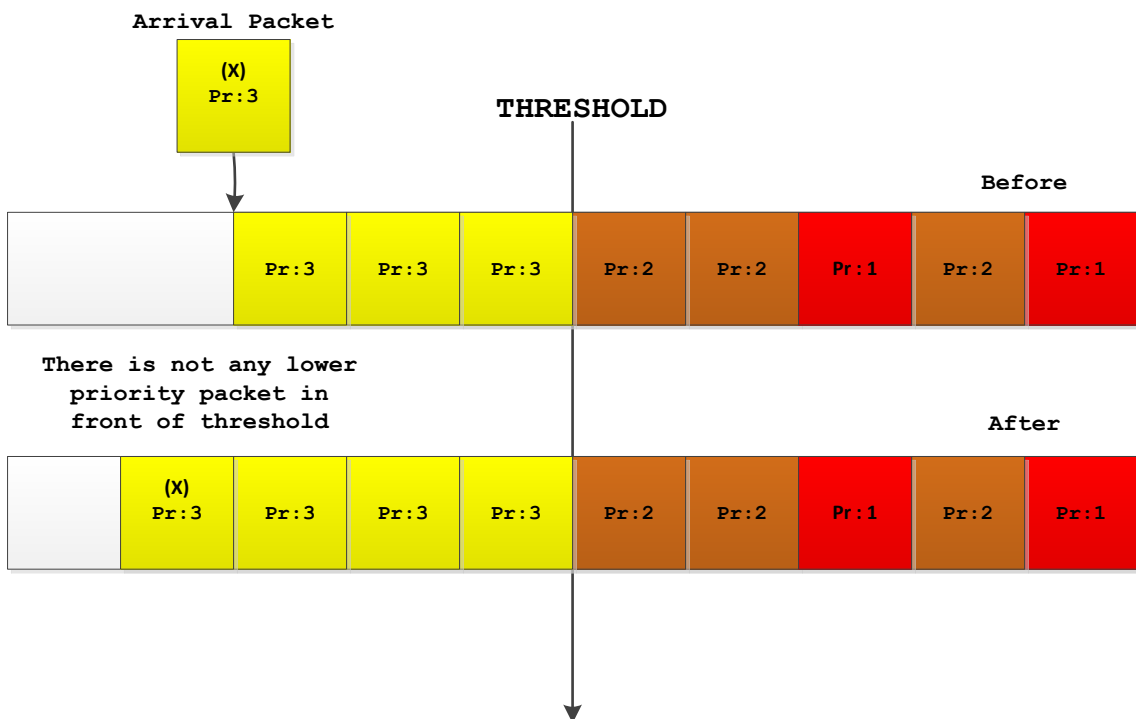


Figure 4.6 – Addition of packet to behind of threshold.

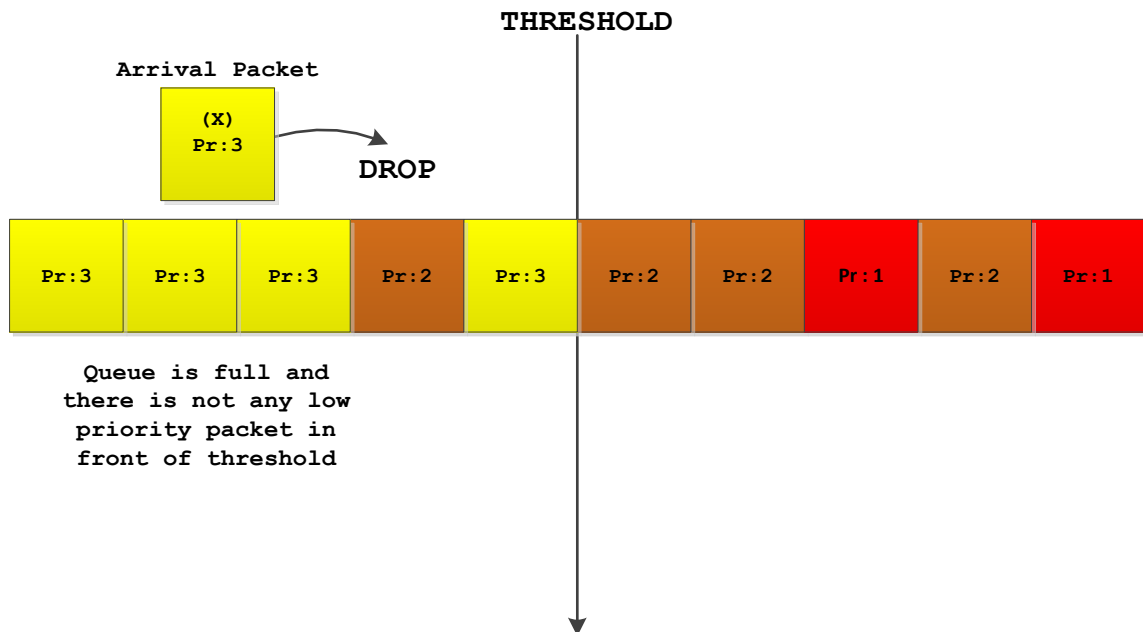


Figure 4.7 – Dropping of lower priority arrival packet.

Appropriate threshold values are assigned and it is looked that there is a place in front of threshold level.

1) If there is a place in front of its threshold value which is determined by arrival packet's priority, the arrival packet is added into end of the Waiting Queue. (see Figure 4.4)

2) If there is not any place in its own threshold, the Waiting Queue is analyzed from its own threshold value ( $k-1$ ) to first element in queue (index=0). It is examined for finding a lower priority packet and if lower priority is found, search process of lower priority is done. Search process is started.

2.1) If there is a lower priority packet than arrival packet's priority level, packet which has low priority is removed from queue and the arrival packet is added instead of lower priority. Packet which is removed from queue is looked for empty place at the capacity of Waiting Queue. (see Figure 4.5)

2.1.1) If there is a place at the Waiting Queue, the packet is added to end of the Waiting Queue. (see Figure 4.6)

2.1.2) If there is not any place at the Waiting Queue, the packet is dropped and added into the Drop Queue. (see Figure 4.7)

2.2) If there is not any lower priority then the arrival packet is looked for the Waiting Queue has place.

2.2.1) If there is a place at the Waiting Queue, the arrival packet is added to end of the Waiting Queue. (see Figure 4.6)

2.1.2) If there is not any place at the Waiting Queue, the arrival packet is dropped and added into the Drop Queue. (see Figure 4.7)

Statistics are collected and the new arrival is called.

## **4.2 Dynamic Multi Threshold Priority Packet Scheduling**

The second proposed algorithm is DMTPS. Packets are placed into the waiting queue according to their priorities and each priority level has its own threshold value. The basic threshold is shown in Figure 4.1. The initial threshold value of each priority level is same as MTPS seen as Figure 4.2.

The main objective of the DMTPS algorithm is to order different priority level packets according to their threshold. The most important operation of the DMTPS algorithm is to locate lower priority packets and replace with higher priority arrival packets at appropriate own threshold value. The only difference with MTPS algorithm is that DMTPS algorithm has dynamic threshold structure.

In DMTPS algorithm, if front of appropriate threshold is full and lower priority packet is found, then it is changed with the higher priority. Higher priority packets are not just ordered with assigned threshold level. If there is not any place at the higher priority packets' assigned threshold, it can be checked at lower threshold levels. For example, Priority-1 (the high priority level) packets are arrived and there is not any place at assigned threshold

(Threshold-1), then it can be looked at lower threshold (Threshold-2) to get in the waiting queue. The higher priority packets (Priority 1 and Priority 2) are checked to their own threshold and if there is not any place to get in, lower threshold levels are used for them. Thus, changing of higher priority level's assigned threshold levels provides that higher level of priority packets to enter to the waiting queue. Dynamic thresholds are used for loss efficiency of higher priority level packets.



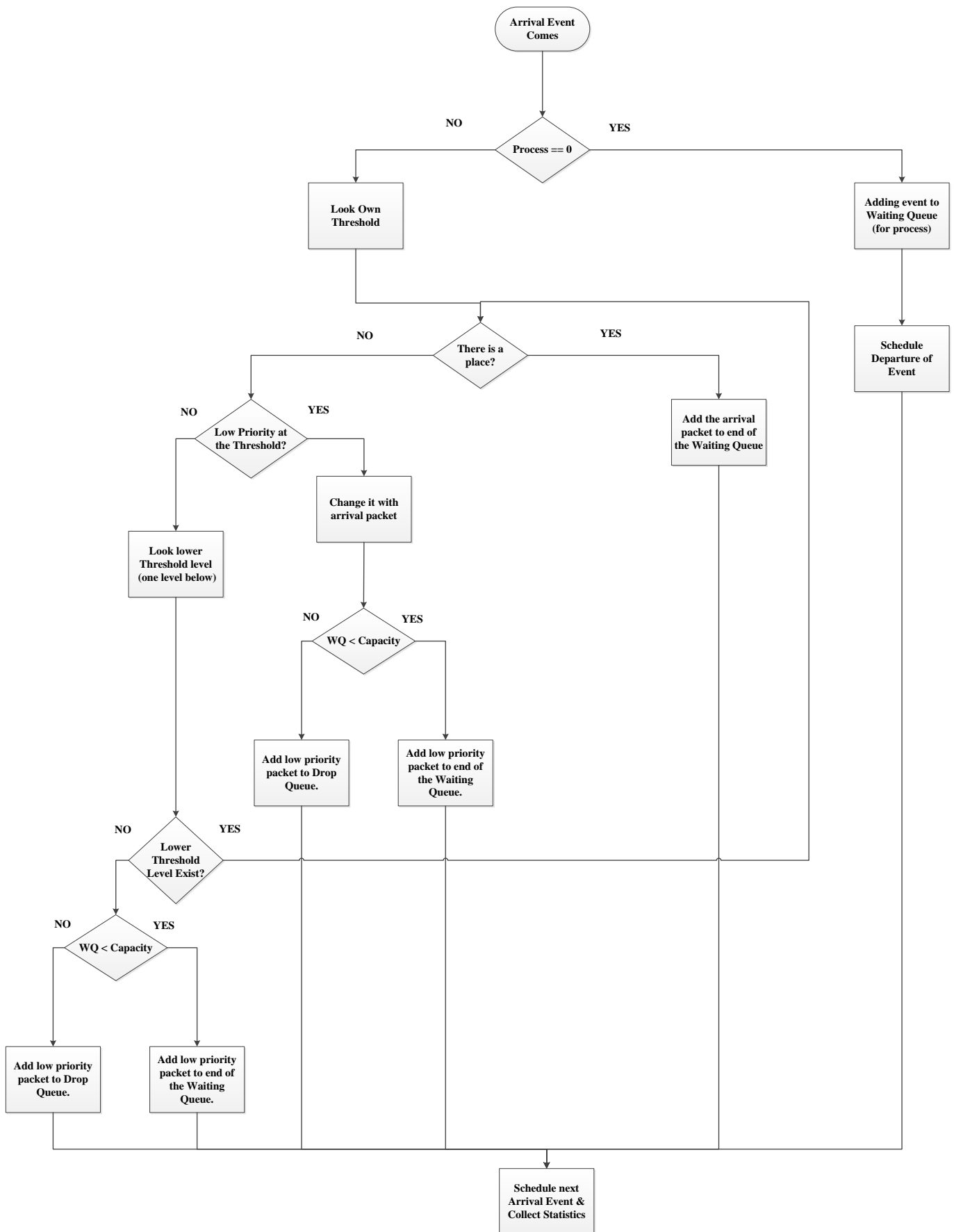


Figure 4.8 - Arrival event of Dynamic Multi Threshold Priority Scheduling algorithm.

Server and waiting queue is full or empty it is controlled. Server is expressed with P and waiting queue is expressed with WQ. Arrival packets go into the system and P and WQ value assigns its next move.

Algorithm of arrival event follows as;

When the Server is empty:

The arrival packet goes into the Server directly without going into the Waiting Queue. Departure of the packet is scheduled and the arrival packet's departure process is started. The new arrival packet is called for Discrete Event System's continuity.

When the Server is full:

Each threshold values are defined values of threshold levels are such as defined at Chapter 4.1. The arrival packet's priority and its own threshold value are analyzed. The packet's appropriate threshold value is assigned by its priority

1) If there is a place to get in front of its threshold value which is determined by arrival packet's priority, the arrival packet is added into end of the Waiting Queue. (see Figure 4.4)

2) If there is not any place in its own threshold, the Waiting Queue is analyzed from its own threshold value ( $k-1$ ) to first element in queue (index=0). It is examined for finding a lower priority packet and if lower priority is found, search process of lower priority is done. Search process is started.

2.1) If there is a lower priority packet than arrival packet's priority, packet which has low priority is removed from queue and the arrival packet is added instead of lower priority.

2.1.1) Packet which is removed from queue is added into end of the Waiting Queue, if the capacity of Waiting Queue is suitable to add. (see Figure 4.5)

2.1.2) Packet which is removed from Waiting Queue is dropped and it is added into to Drop Queue, if capacity of Waiting Queue is full.

2.2) If there is not any lower priority packet in front of assigned threshold, then it is looked at any lower threshold levels for arrival packet's priority level. It is looked at lower threshold level according to priority level of the packet.

2.2.1) If there is any lower threshold level for packet's priority level, then back to Step 2 with value of lower threshold level.

2.2.2) If there is not any lower threshold level for packet's priority level, then the arrival packet is dropped or added end of the Waiting Queue according to concurrency of the Waiting Queue.

2.2.2.1) The arrival packet is added into end of the Waiting Queue, if the capacity of Waiting Queue is suitable to add. (see Figure 4.6)

2.2.2.2) The arrival packet is dropped and it is added into to Drop Queue, if capacity of Waiting Queue is full.

Statistics are collected and the new arrival packet is called.

### **4.3 Dynamic Multi Threshold Priority Packet with Urgency Packet Scheduling**

DMTPUS algorithm is assumed as similar with MTPS and DMTPS which are declared at Chapter 4.1 and Chapter 4.2. The initial threshold value of each priority level is same as MTPS. Dynamic multi -threshold structure of the DMTPUS algorithm is also similar with the DMTPS algorithm. Also, priority level status and threshold placement is the same as the DMTPS algorithm.

The algorithm provides the order of each priority level with its threshold. The most important operation of the DMTPUS algorithm is to locate lower priority packets and replace with higher priority arrival packets at appropriate own threshold value same as DMTPS algorithm. The only difference between DMTPS and DMTPUS algorithm is that packets have urgent status which specifies packets should be processed quickly at DMTPUS algorithm. It is an efficient technique for scheduling of packets which is urgent and should be processed immediately. Urgent status is defined as a field ( $u$ ) and designed for solving some problems.

Excessively losing of lower priority data which must be processed quickly encounters some problems. Data losses may occur continuously and may not be processed at other scheduling algorithms. Real time data are upgraded as urgent and can be processed with privileges. Table 7.1 shows that the DMTPUS algorithm provides precedence to packets which have urgent status.

Table 4.1 - Urgent Status

Priority Level	Urgent	Not Urgent
Priority 1	Marked as the Most Valuable Packet ( The highest priority level and higher than Priority 1)	Priority 1
Priority 2	Priority 1	Priority 2
Priority 3	Priority 2	Priority 3

Packets which have urgent status are evaluated as follows.

Priority 1(The highest priority level) packet with Urgent Status – It is marked as most valuable packet. This packet is added to head of the waiting queue because it should be processed immediately. All elements of the waiting queue shift one step back with the addition of the most valuable packet. Figure 4.10 shows an example of the behavior of system when the high priority level packet with urgent status arrives.

Priority 2(The medium priority level) packet with Urgent Status – Its priority level is upgraded and marked as Priority 1 packet. This packet then behaves as a packet which has the higher priority level.

Priority 3 (The lowest priority level) packet with Urgent Status - Its priority level is upgraded and marked as Priority 2 packet. This packet then behaves as a packet which has higher priority level.

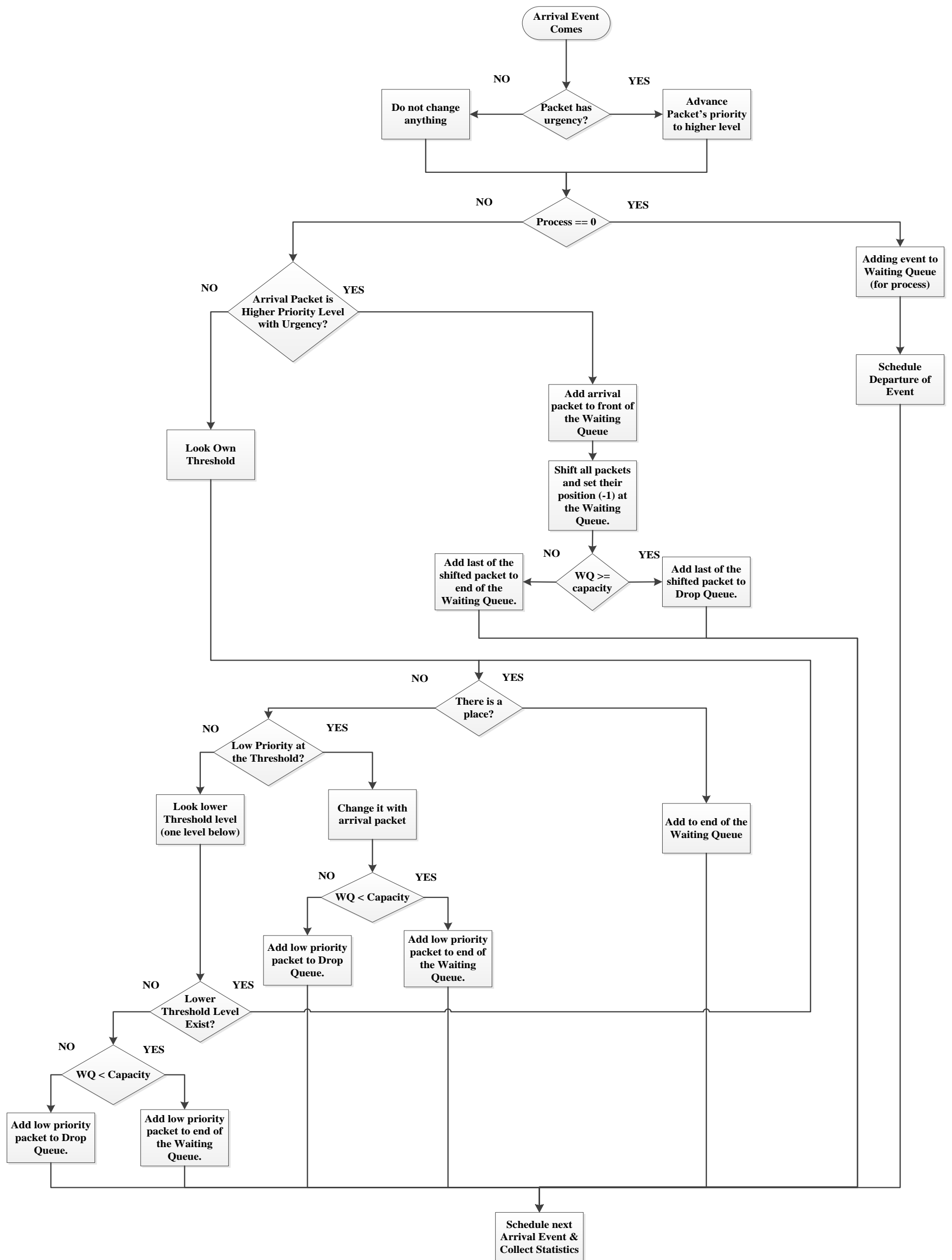


Figure 4.9 - Arrival event of Dynamic Multi Threshold Priority with Urgency Scheduling.

Server and waiting queue is full or empty it is controlled. Server is expressed with P and waiting queue is expressed with WQ. Arrival packets go into the system and P and WQ value assigns its next move. Firstly, urgency is determined according to Table 7.1 when the arrival packet goes into the system.

Algorithm of arrival event follows as;

When the Server is empty:

The arrival packet goes into the Server directly without going into the Waiting Queue and the arrival packet's departure process is started. The new arrival packet is called for Discrete Event System's continuity.

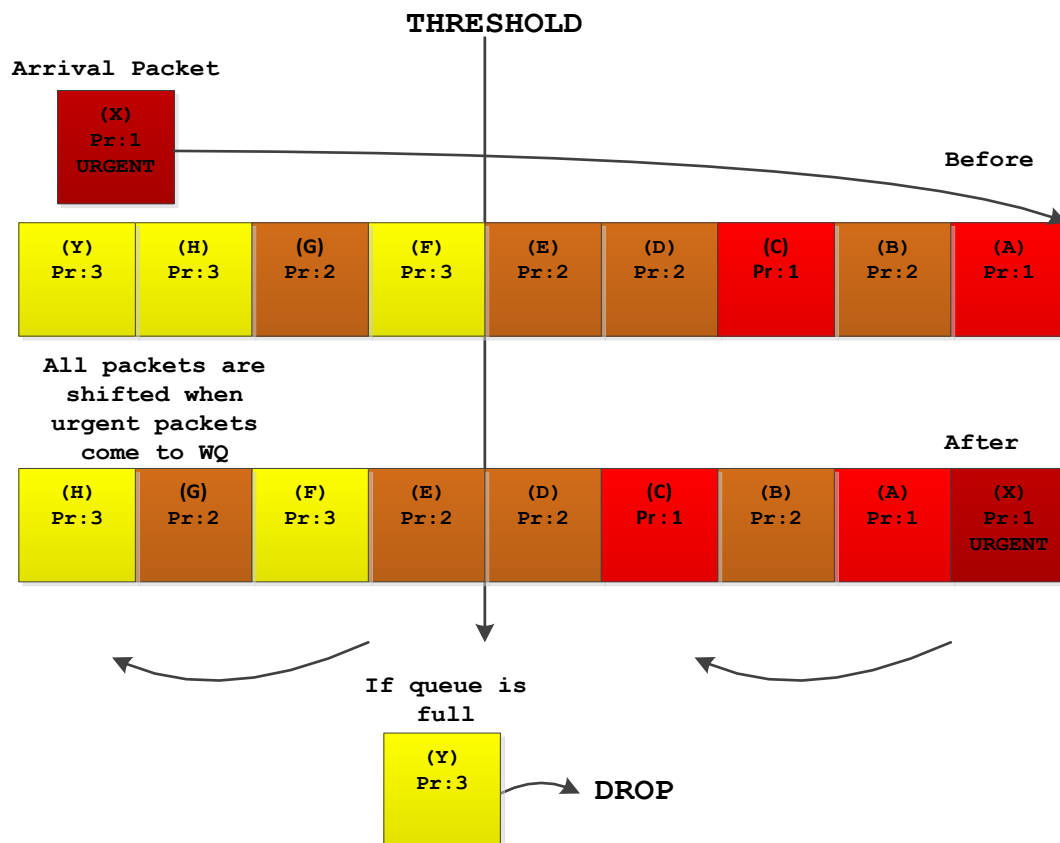


Figure 4.10 - Sliding with arrival of the highest priority packet enters to system.

When the Server is full :

Each threshold capacities are defined. The arrival packet's priority and its own threshold value are analyzed. The packet's appropriate threshold value is assigned by its priority.

1) Checking if the arrival packet has the highest priority level with urgency status.

1.1) If the arrival packet has the highest priority level with urgency status, then the arrival packet is added the head of the Waiting Queue. The elements of the Waiting Queue are shifted one step backward and end member of the Waiting Queue is appropriate for Waiting Queue's capacity is checked. (see Figure 4.10)

1.1.1) If capacity has element to add, it is added to end of the Waiting Queue.

1.1.2) If capacity is not appropriate to add, the end member of the queue is removed from the Waiting Queue and it is added to Drop Queue.

1.2) If the arrival packet has not highest priority level with urgency then, it is checked if assigned threshold value of packet has place to get in. The packet's appropriate threshold value is assigned by its priority

1.2.1) If there is a place to get in front of its threshold value which is determined by arrival packet's priority, the arrival packet is added into end of the Waiting Queue. (see Figure 4.4)

1.2.2) If there is not any place in its own threshold, the Waiting Queue is analyzed from its own threshold value ( $k-1$ ) to first element in queue (index=0). It is examined for finding a lower priority packet and if lower priority is found, search process of lower priority is done. Search process is started.

1.2.2.1) If there is a lower priority packet than arrival packet's priority, packet which has low priority is removed from queue and the arrival packet is added instead of lower priority. It is looked at concurrency of the Waiting Queue.

1.2.2.1.1) Packet which is removed from queue is added into end of the Waiting Queue, if the Waiting Queue has capacity.

1.2.2.1.2) Packet which is removed from queue is dropped and it is added Drop Queue, if the Waiting Queue has not capacity. (see Figure 4.5)

1.2.2.2) If there is not any lower priority packet in front of assigned threshold, then it is looked at any lower threshold levels for arrival packet's priority level.

1.2.2.2.1) If there is a lower threshold level for packet's priority level, then back to Step 1.2.2 with value of lower threshold level.

1.2.2.2.2) If there is not any lower threshold level for packet's priority level, then It is looked at concurrency of the Waiting Queue.

1.2.2.2.2.1) Packet which is removed from queue is added into end of the Waiting Queue, if the Waiting Queue has capacity.

1.2.2.2.2.2) Packet which is removed from queue is dropped and it is added Drop Queue, if the Waiting Queue has not capacity. (see Figure 4.5)

Statistics are collected and the new arrival packet is called.

#### **4.4 Implementation of the Algorithms**

Java programming is a widely available programming language that has been used extensively in simulation. The Java ® programming language is an object-oriented and class based language that used for running applications [11]. It does not, however, provide any facilities directly aimed at aiding the simulation analyst, who therefore must program all details of the event-scheduling/time-advance algorithm, the statistics gathering capability, the generation of samples from specified probability distributions, and the report generator. [10]

Components of the simulation which written in Java are:

*CLOCK* – It is variable that describe simulation time.

*Initialization method* – It is a method that beginning of program system state is described.



*Minimum-Time Event method* – Time of events are sorted in the Future Event List, which has smallest time it is got from Future Event List, it is called as imminent event.

*Event Methods* – System state and cumulative statistics are updated when event occurs.

*Random-variate generators*-It is a method that samples are generated from intended probability distributions.

*Main Program* – The event-scheduling algorithm's overall control is aimed to be maintained. .

*Report Generator* – It is a report generator that summary of statistics is calculated from cumulative statistics.

Simulation is performed according to Discrete Event System Simulation (see Chapter 2). Concepts are generated with Concepts in Discrete Event System at Chapter 2.4.1. Event-scheduling simulation is originated from Chapter 2.4.2. The only difference with event-scheduling simulation is that packets are scheduling in this simulation.

Event-scheduling simulation program's overall structure is as below.

Step 1 - Simulation is started at main program and input parameters are obtained.

Step 2 - Initialization method is called. CLOCK and cumulative statistics are equaled to 0. Initial events are generated and placed on Future Event List.

Step 3 - Time-advance subroutine is called at main program.

Step 4 - Imminent event's time is advanced to CLOCK.

Step 5 - Appropriate event subroutine is called at main program.

Step 6 - At event subroutine, event is executed and system state and entity attributes are updated. Cumulative statistics are gathered. Future events are generated and to be placed on Future Event List.

Step 7 - If simulation is not over, it is continued and passed to Step 3. If simulation is over, summary statistics are calculated and report is generated.

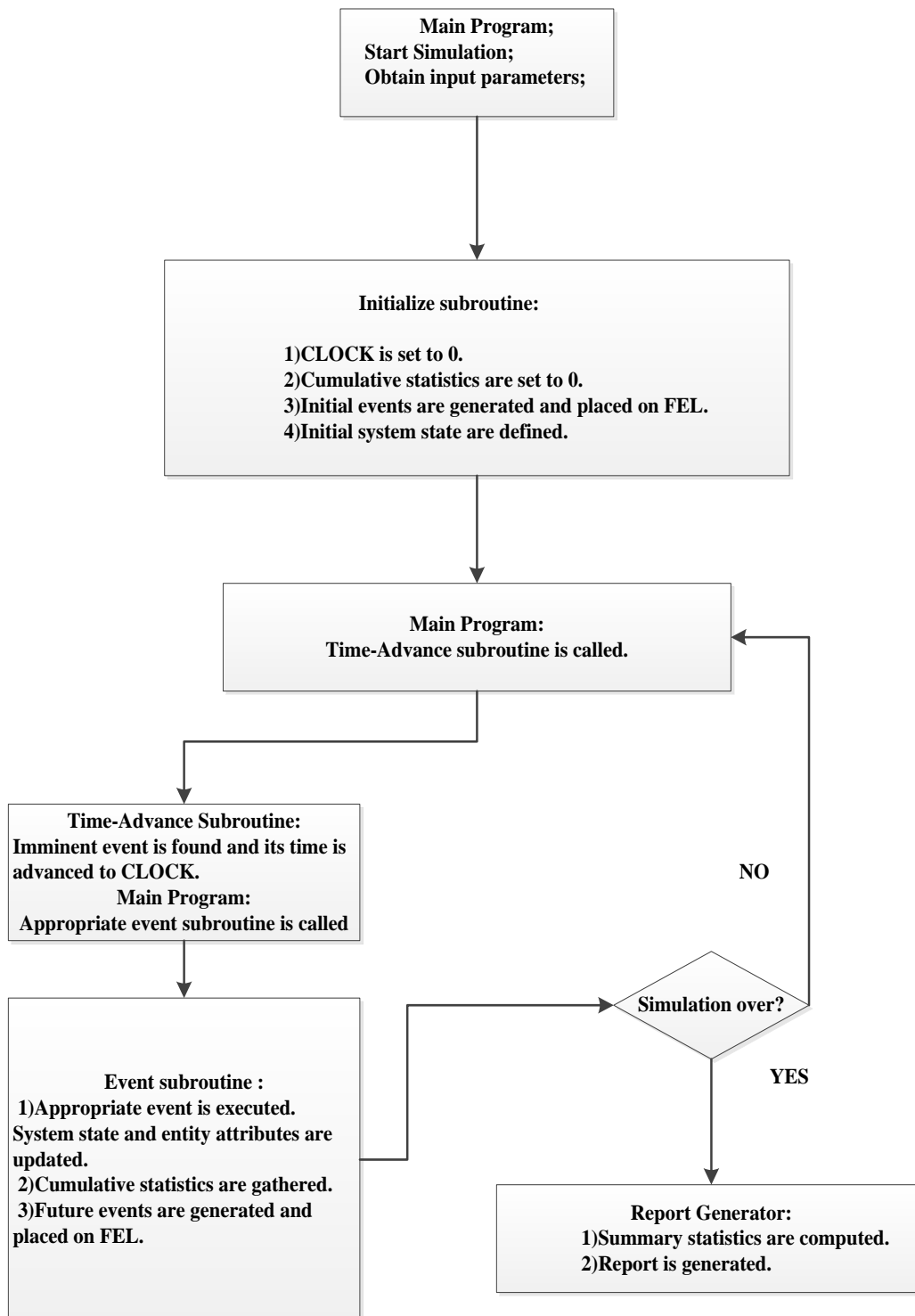


Figure 4.11 - Event scheduling program's structure.

Table 4.2 - Definition of variables.

Variables	Description
<b>System State</b>	
<ul style="list-style-type: none"> <li>○ WQ</li> </ul>	Number of packets queued at current simulated time.
<ul style="list-style-type: none"> <li>○ DQ</li> </ul>	Number of packets which are dropped at current simulated time.
<ul style="list-style-type: none"> <li>○ Process(P)</li> </ul>	Number being served at current simulated time. (0=empty, 1=busy)
<b>Entity attributes and sets</b>	
<ul style="list-style-type: none"> <li>○ PacketList</li> </ul>	Queue of packets in system.
<ul style="list-style-type: none"> <li>○ ThresholdList</li> </ul>	The list of threshold per priority level.
<ul style="list-style-type: none"> <li>○ ArrivalListPerPriority</li> </ul>	List of arrival packets per priorities.
<ul style="list-style-type: none"> <li>○ DropListPerPriority</li> </ul>	List of drop packets per priorities.
<ul style="list-style-type: none"> <li>○ DelayListPerPriority</li> </ul>	List of delay time's per priority level.
<ul style="list-style-type: none"> <li>○ TotalUrgentPerPriority</li> </ul>	List of packets which have urgent status per priority level.
<ul style="list-style-type: none"> <li>○ TotalUrgentDropPerPriority</li> </ul>	List of packets which have urgency is dropped per priority level.
<b>Future Event List</b>	
<ul style="list-style-type: none"> <li>○ FutureEventList</li> </ul>	Priority-ordered list according to event's data of pending events.
<b>Activity durations</b>	
<ul style="list-style-type: none"> <li>○ MeanInterarrivalTime</li> </ul>	The total inter-arrival time between the previous packet's arrival and the next arrival.
<ul style="list-style-type: none"> <li>○ MeanServiceTime</li> </ul>	The service time of the most recent packet to begin service.
<b>Input parameters</b>	
<ul style="list-style-type: none"> <li>○ MeanInterarrivalTime</li> </ul>	Mean interarrival time(ex:10 ms)
<ul style="list-style-type: none"> <li>○ MeanServiceTime</li> </ul>	Mean service time(ex:9 ms)
<ul style="list-style-type: none"> <li>○ Packets</li> </ul>	The stopping criterion number of packets to be served
<ul style="list-style-type: none"> <li>○ Capacity</li> </ul>	The capacity of the queue.

○ Urgency	Urgent status of the packet.
○ ID	ID of the packet.
○ Priority	Priority of the packet.
<b>Simulation variables</b>	
○ CLOCK	The current value of simulated time.
<b>Statistical Accumulators</b>	
○ LastEventTime	Time of occurrence of the last event.
○ TotalBusy	Total busy time of server.
○ NumberOfDepartures	Number of departure packets.
○ SumResponseTime	Sum of packets response times who are departed from the system.
○ TotalArrival	Total arrival packets.
○ TotalDrop	Total drop of packets.
○ TotalDelay	Total delay of packets.
○ TraceID	It is a value that it is assigned lower priority packets location at the queue.
○ Start	Starting time of the departure packet's process.
○ Finish	Finishing time of the departure packet's process.
○ LastProcessTime	Last departed packet's process time of the simulation.
○ IdleTime	Total Idle time of the simulation.
○ TotalUrgent	Total packets which have urgent status.
<b>Summary Statistics</b>	
○ Utilization	$(\text{totalBusy}/\text{CLOCK})$ , busy time proportion according to system time
○ TotalLossRatio	$(\text{totalDrop}/\text{totalArrival})$ , the total loss ratio
○ LossRatioPerPriority	$(\text{DropPerPriority}/\text{ArrivalperPriority})$ , the loss ratio per packet's priority.

Table 4.3 - Definition of methods

<b>Methods</b>	<b>Description</b>
Initialization	Method of initialization
ArrivalMethod	Arrival event's execution method.
DepartureMethod	Process event's execution method.
ReportGeneration	Generator of report.

Table 4.4 - Definition of functions

<b>Functions</b>	<b>Description</b>
exponential	Generating values from exponential distribution.
compare	Comparing events per priority.
traceQueue	Finding lower priority packet at the queue.
frontOfThreshold	Looking packet's own threshold per priority.
thresholdControl	Controls threshold and changing values of threshold per priority level.
randomUrgency	Generating randomly urgency status of packet's.
urgencyCheck	Checking packet have urgency or not and evaluating accordingly.
shift	Shifting all elements when packet which quick response packet is placed head at the Queue.

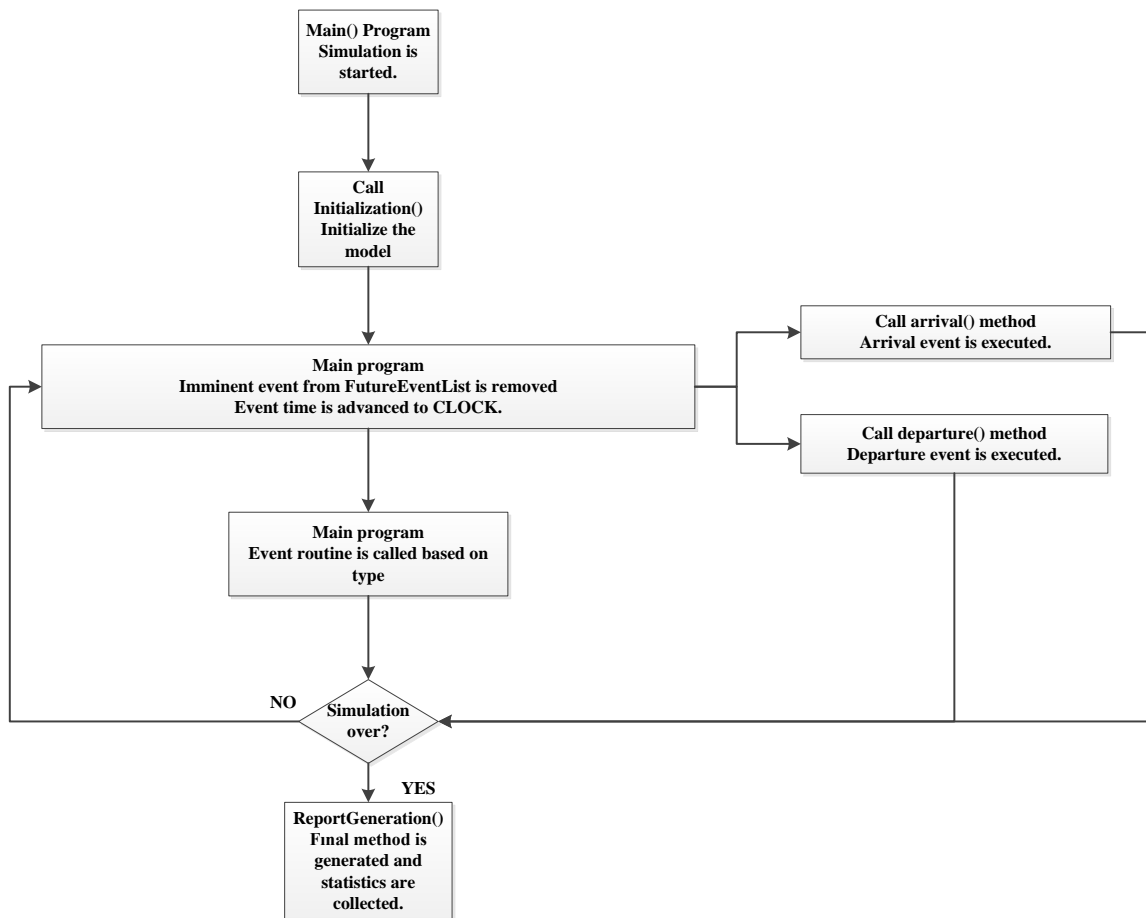


Figure 4.12 - Structure of simulation.

Step 1 – Simulation is started.

Step 2 – Model is initialized.

Step 3 – Imminent event from Future Event List are removed.

Step 4 – Event routine based on type is called.

Step 5 – If type is Arrival, then arrival event is called. If it is Departure, departure event is executed.

Step 6 – If simulation is progressed, then step-3 is recalled and the simulation goes on. If simulation is over, then the report generation method is called.

#### **4.4.1 Main Program**

First of all, all variables and sets are defined to system. Entity attributes, sets and classes are declared at top of the main class (PacketList, FutureEventList, TotalArrival, TotalDrop etc.) At the main class, some attributes are defined such as meanServiceTime. Sets and lists are defined at the main class. Packets and capacity values are demanded from the user. Threshold values are initialized for priority levels. The threshold values are declared at Chapter 4.1. Then, initialization() method is called. After calling of initialization method, loop of the simulation is started. The loop of the program is run until total packets which entries from user have created. Events are called from Future Event List. Future Event List is sorted according to event's time with ascending order. It is polled and removed from the Future Event List. The event's time is advanced to CLOCK time. The event's type specifies the function. If it is an arrival event, arrival method is called. If it is not, departure event is called.

When all of packets are produced exactly, Report Generation method is called and the simulation is over.

##### ***4.4.1.1 Generating Activity Times***

Generation of activity times are produced by statistical distributions. Arrival, service, priority and deadline times are generated by some of circumstances.

##### ***4.4.1.1.1 Simulating Arrival Time***

Generation of arrival time values is done with the exponential distribution. Exponential distribution is a continuous probability distribution group in probability theory and statistical science theory. Exponential distribution is a family of continuous probability distributions. It describes the time between events in a Poisson process, a process in which events occur continuously and independently at a constant average rate [12].

Definition of exponential distribution's variables:

- mean - mean value of an event
- seed - used for initializing a random number generator.

Equation of exponential distribution is shown as;

$$\text{exponential} = -\text{mean} \times \log(\text{seed})$$

Value of event's time is assigned with exponential distribution of mean arrival time. For example, in these simulations, there are three priority levels and each priority level has its mean arrival time. Each priority levels arrival time are separately calculated with exponential distribution.

#### **4.4.1.1.2 Simulating Service Time**

Generation of a service time values are done with the exponential distribution. Value of event's time is assigned with exponential distribution of mean service time. For example, in these simulations, 9.5 and 9.9 are assigned to mean service time value.

#### **4.4.1.1.3 Generating Priority**

The priorities are created from the finite number of possible values. Priority of an event is assigned at starting of simulation. For example, in these simulations, priority value of the event may be assigned between 1 and 3. Value of 1 specifies the highest priority. Value of 3 specifies the lowest priority. The other values specify their priorities in a descending order.

#### **4.4.1.1.4 Generating Random Urgency**

Generation of a random urgent status operation is done with random number generator. For example, each packet's urgent status possibility is %20.

### **4.4.2 Initialization**

Variables are defined. Initialize of CLOCK, Process, LastEventTime, TotalBusy, NumberOfDepartures, WQ, DQ etc. values are assigned as 0. meanArrivals for different priority levels are assigned with  $ma$ ,  $ma*2$ , and  $ma*3$ .  $ma$  refers to meanInterArrival time and its values is 110/6. The meanServiceTime value is assigned to 9.5 and 9.9 for different runs.



Event
-ID
-Type
-Arrival Time
-Service Time
-Priority
-Data
-Urgency

Figure 4.13 - Event Class

The first arrival events are created after initialize the variables and sets. Event has seven attributes. These are id, type, arrivalTime, serviceTime, priority, data and urgency. Type is denoted with “A” and “D”. “A” specifies the arrival event. “D” specifies the departure event. The arrivalTime and serviceTime are generated with function of exponential distributions. Priority is generated when first arrival events are generated. Data is a key for Future Event List, event list is ordered by event’s data. Urgency is generated from random numbers. First arrival events for each priority are generated with these values. For each priority level, one arrival event is added to Future Event List. There are two list which are Future Event List and Waiting Queue list, Future Event List deals with events and waiting queue list deals with packets.

#### 4.4.3 Arrival Method

The value of process ( $P$ ) is important for flow of arrival method. If  $P$  value is equal to 0, then it refers the server is empty and the packet goes directly to process section (server) to start its departure. If  $P$  value is equal to 1, it refers that server is busy. If  $P$  is equal to 0, then the packet is added to server and its departure is scheduled. The values of events are added to Future Event List with type of departure. It is done with Type is changed to “D” and data field is changed to  $t+s^*$  and added FEL and  $P$  value is equaled to 1.

Packet is taken from waiting queue and the scheduling process of packet is started.

If server is busy, then it is looked at capacity has empty place. If capacity has an empty place, the packet is added to Waiting Queue. The next arrival event is scheduled when addition operation completes. If capacity has not any empty place, the event is behaved by

algorithm's characteristics. The First-In-First-Out algorithm, it is dropped. The Priority Queue algorithm, it is looked at the last event of the Waiting Queue (WQ). If its priority is lower than the arrival packet, the arrival packet is added into Waiting Queue. The end of these operations the new arrival event is generated. The processed arrival event's priority specifies the new arrival event's priority and same priority is assigned to the new arrival event. It is added to Future Event List with the appropriate attributes of its priority level.

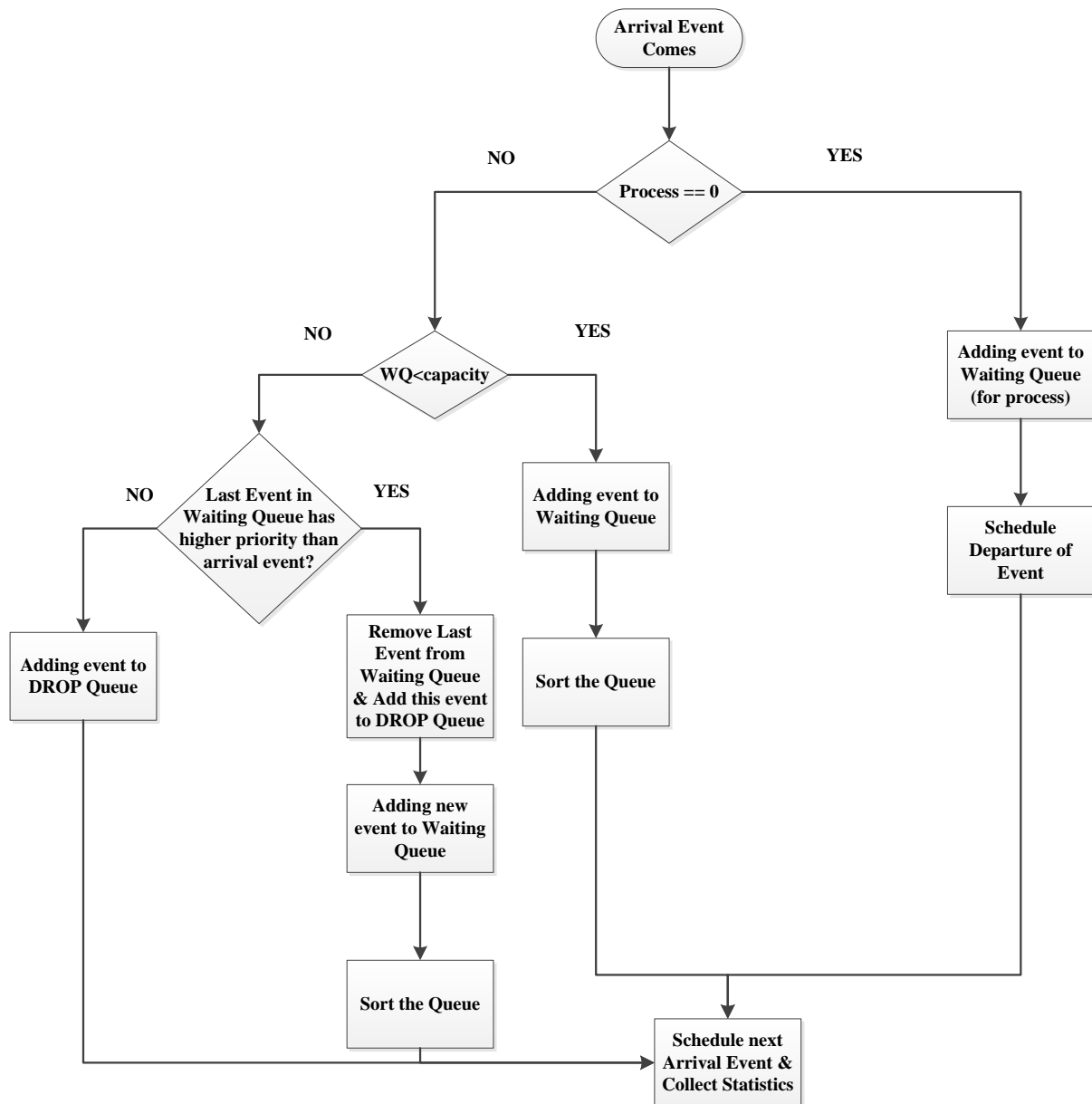


Figure 4.14 - Arrival event of Priority Queue scheduling algorithm.

#### 4.4.4 Departure Method

The value of (WQ) is important for departure method. If departure method is called, then it means departure of a packet is finished.

If WQ value is not equal to 0, then WQ value is reduced by 1 and the appropriate event from WQ is taken. It is taken departure packet from the waiting queue and departure of event is scheduled. After that, the new departure packet from Waiting Queue is generated and statistics are collected.

If WQ values is equal to 0, then process value ( $P$ ) is assigned as 0 and it refers that server is empty. Statistics are collected and returns to time-advance method.

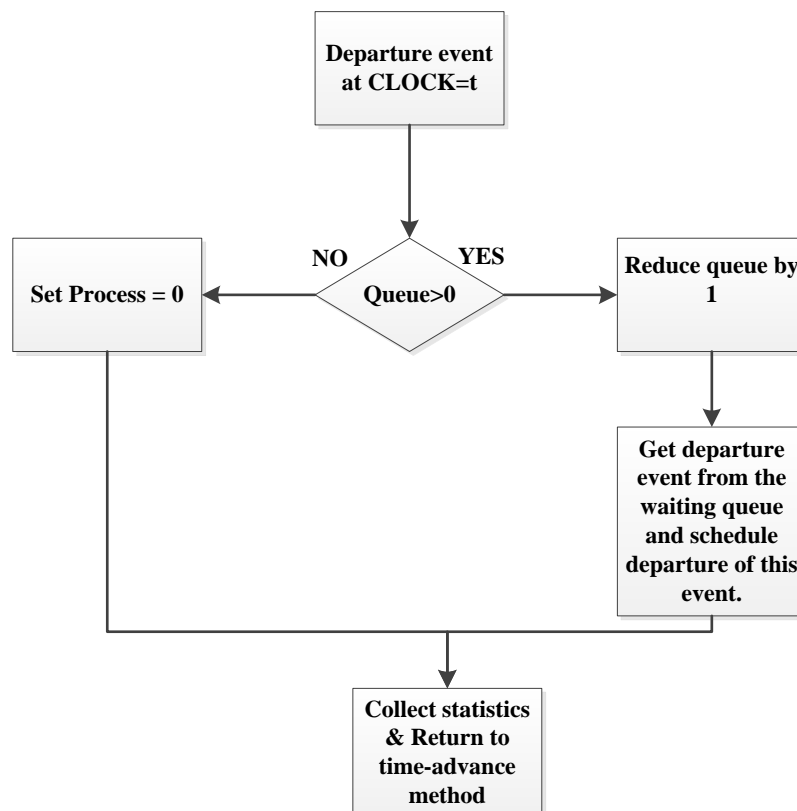


Figure 4.15 - Departure event of Priority Queue scheduling algorithm.

#### 4.4.5 Report Generation

For analyze and evaluate of this values and its reaction, report generation method is created. At the report generation method, the statistics are shown to the user.

Number of departure packets, arrival and dropped packets, last event time, total busy time, sum response time, server utilization, total arrival packets per priority, total dropped packets per priority and their loss ratios, total delay time and total delay time per priority statistics are shown to the user.

## 5 ANALYSIS OF MULTI-THRESHOLD SCHEDULING ALGORITHMS

The simulation of proposed algorithms is performed using Discrete Event System Simulation in Java. The simulation results are illustrated with figures which show the loss ratio and the average delay time.

The throughput value of a system is a factor that has impact on value of loss ratio and delay time results. The value of loss ratio increases with the increase of throughput value. Capacity of waiting queue is a factor - such as - that it has very high impact on value of loss ratio. The total loss ratio value decreases with the increase of capacity.

Initial input values of system and their denotations are;

- Number of Packet (packets)
- Capacity of Waiting Queue (capacity)
- Arrival Rate ( $\lambda$ )
- Service Rate ( $\mu$ )

Arrival and service time values are generated with exponential distribution.

The throughput value of the system is specified with dividing of arrival rate to process rate. The throughput value of system is denoted with  $\alpha$  and the equation of  $\alpha$  is as;  $\alpha = \frac{\lambda}{\mu}$

.

System stability depends on the following condition.  $\alpha$  must be smaller than 1 if the system continues its stability. Explosion at the system may occur without applying this condition.

Number of packets is assigned to  $100 \times 10^6$  for all experiments.

Throughput value of a system is assigned as 0.95 and 0.99. Capacity of waiting queue is assigned with values between 20 and 100. Threshold level's values are described at Chapter 4.1.  $k$ ,  $m$  and  $n$  values are assigned as 60, 80 and 100 respectively. Urgent status which is valid for DMTPUS algorithm declared at Chapter 4.3. Generation of a random urgent status operation is done with an arbitrary discrete distribution. Value of  $u$  is assigned as 0.2.

Results of the selected priority level are shown in the figures. The analyzing of results is examined with four phase. First of two phases are about to comparison of loss ratio results. Third and fourth phases are about to analysis of delay time. Total loss ratio, total delay time, loss ratio and average delay time per priority level is analyzed.

*First Phase:* The total loss ratio value of these algorithms is experimented.

*Second Phase:* Each value of loss ratio per priority levels are experimented by these algorithms.

*Third Phase:* The total delay time of these algorithms is experimented.

*Fourth Phase:* Each value of average delay time per priority levels are experimented by these algorithms.

### **5.1 First Phase – Total Loss Ratio Analysis**

First of all, the total loss ratio of the each algorithm is analyzed. The aim of this experiment is to find and compare loss ratios according to capacity and  $\alpha$  value.

Capacity of waiting queue is assigned by many values for following up change of loss ratio and delay times. The initial value of Waiting Queue capacity is 20 and it is increasing up to 100. Expected arrival time is 10ms (millisecond) and arrival rate ( $\lambda$ ) is assigned to  $\frac{1}{10}$ . Expected arrival time is experimented for 9.5ms and 9.9ms values. Service rate ( $\mu$ ) is assigned to  $\frac{10}{95}$  and  $\frac{10}{99}$  respectively.

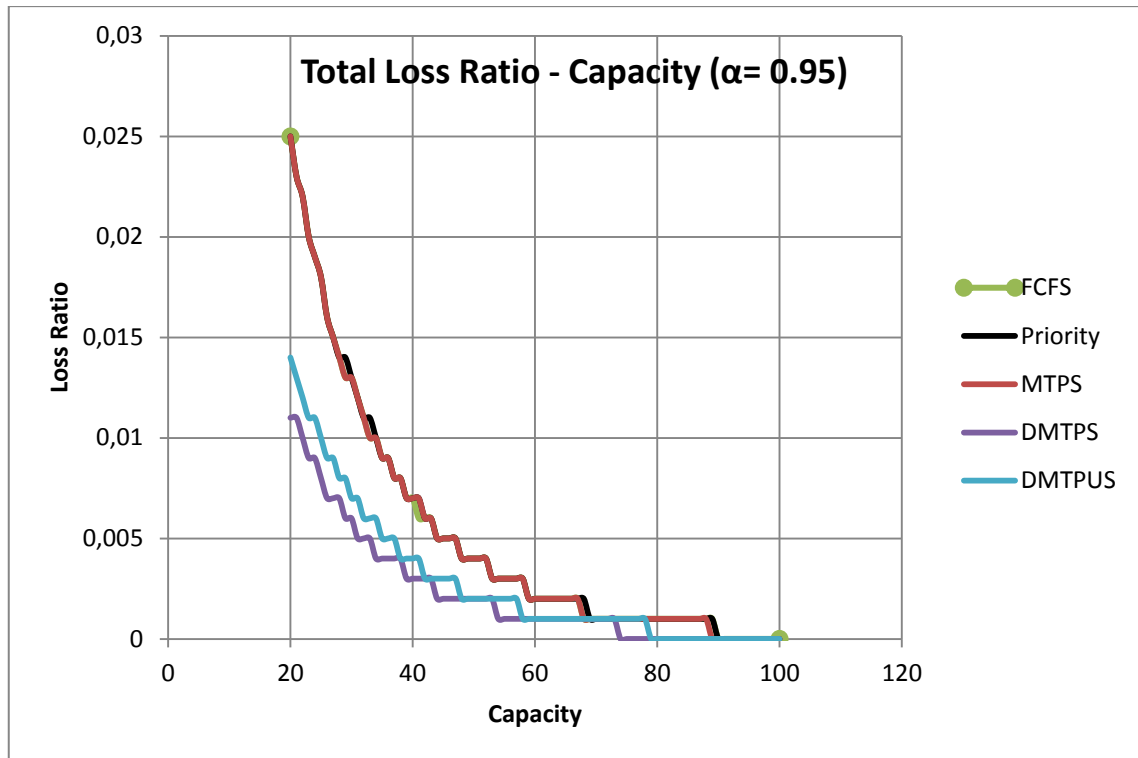


Figure 5.1 - The total loss ratio - capacity graph for all algorithms, when throughput value is 0.95.

FCFS, Priority Queue scheduling and MTPS have similar results when throughput value is 0.95. Their initial loss ratio is about 0.025 and it indicates that decrease of loss ratio value to 0 with increasing of capacity. Specially, it is observed that dynamic threshold packet scheduling algorithms (DMTPS and DMTPUS) have less loss ratio then other algorithms. Throughput value ( $\alpha$ ) is changed 0.95 to 0.99 and the results are analyzed.

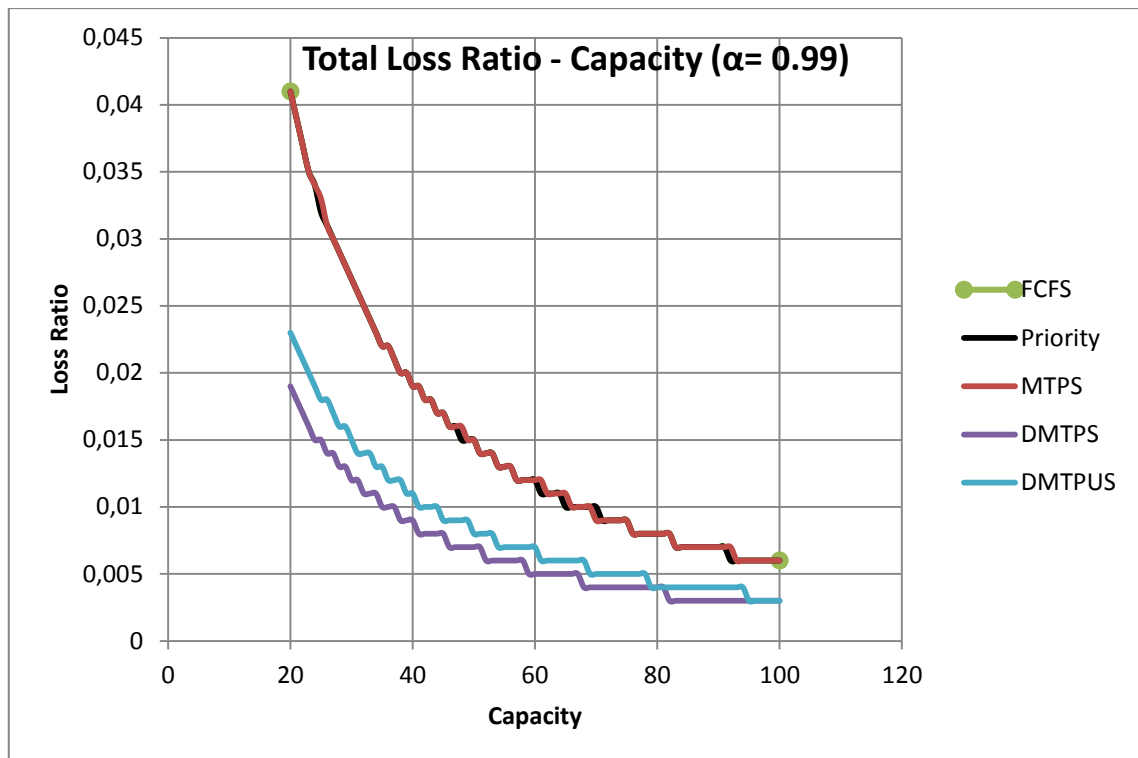


Figure 5.2 - The total loss ratio - capacity graph for all algorithms, when throughput value is 0.99.

The loss ratio of all scheduling algorithms increase while throughput value is changed 0.95 to 0.99 according to Figure 5.1. FCFS, Priority Queue Scheduling and Multi Threshold Fixed-Priority Scheduling have similar results and dynamic threshold scheduling algorithms (DMTPS and DMTPUS) have less loss ratio then other algorithms as seen at Figure 5.2.

## 5.2 Second Phase – Loss Ratio Analysis for each Priority Level

Loss ratio of the each priority level is analyzed at these algorithms. The aim of this experiment is to find and compare loss ratios according to capacity and  $\alpha$  value. Analyzing of results is experimented with changing of these values. The value of arrival time is stable and it is 10ms (millisecond) and arrival rate ( $\lambda$ ) is assigned to  $\frac{1}{10}$ . Capacity of waiting queue and throughput value of the system  $\alpha$  are assigned different values for analyzing different conditions.



### Capacity of waiting queue is 50 and $\alpha$ is 0.95

Capacity of waiting queue is assigned to 50. Expected service time is experimented for 9.5ms and Service rate ( $\mu$ ) is assigned to  $\frac{10}{95}$ .

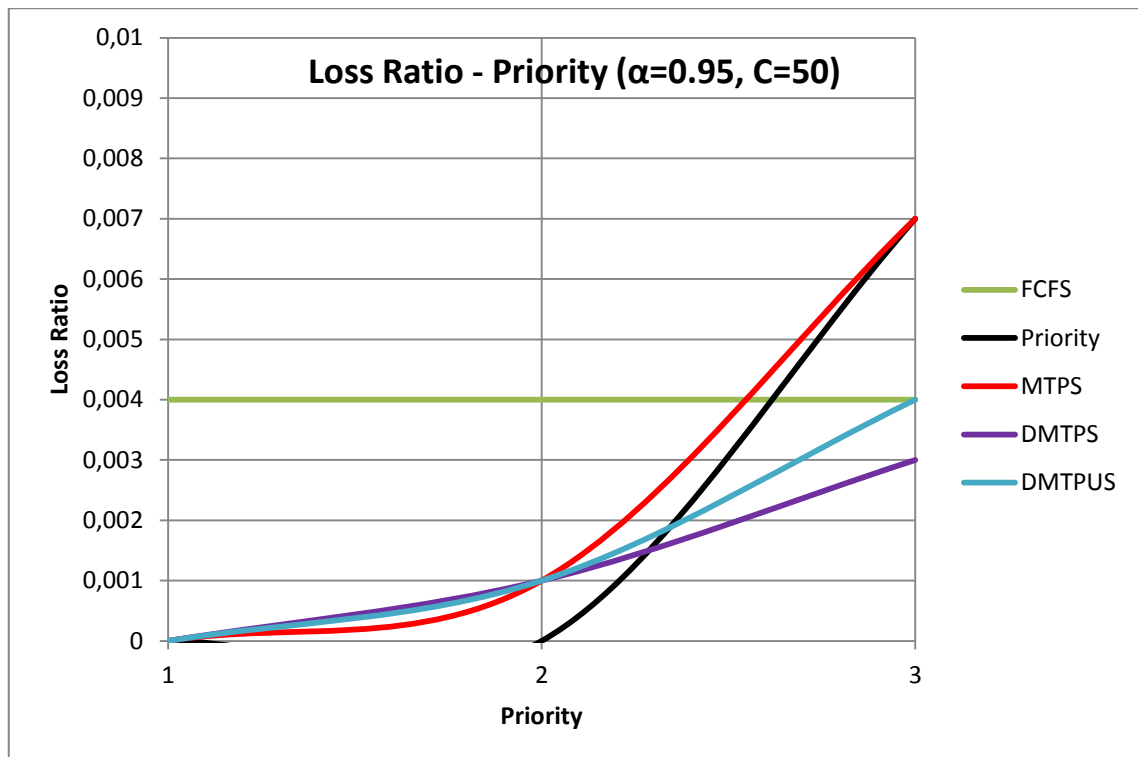


Figure 5.3 - Loss ratio - Priority graph for all algorithms, when throughput value is 0.95 and capacity is 50.

FCFS: The loss ratio of packets for each priority level is same (0.004)

Priority Queue: The loss ratio of packets for each priority level is not same as FCFS. Packet which has higher priority as numerated Priority 1 and 2 level's loss ratio is 0. The lowest priority packets which are numerated level 3's loss ratios are 0.007. Specially, the loss ratio of the lowest priority packets is very high.

MTPS: The loss ratio of lower priority packets increases with logarithmic structure. The highest priority packets' loss ratio is 0 and value of loss ratio increases with decreasing of priority. The lowest priority packets' loss ratio is 0.007.

**DMTPS:** Packets' loss ratio values for higher priority levels are similar with MTPS. The significant change is observed at the lowest priority level. The value of loss ratio for priority level 3 is 0.003 and it is the lowest value according to comparison of loss ratio of lowest priority levels for all algorithms.

**DMTPUS:** Logarithmic increase of loss ratio is observed for descending priority level. It has similar loss ratio values such as DMTPS algorithm but the lowest priority packets have slightly more loss.

### Capacity of waiting queue is 100 and $\alpha$ is 0.95

Capacity of waiting queue is assigned to 100. Expected service time is experimented for 9.5ms and Service rate ( $\mu$ ) is assigned to  $\frac{10}{95}$ .

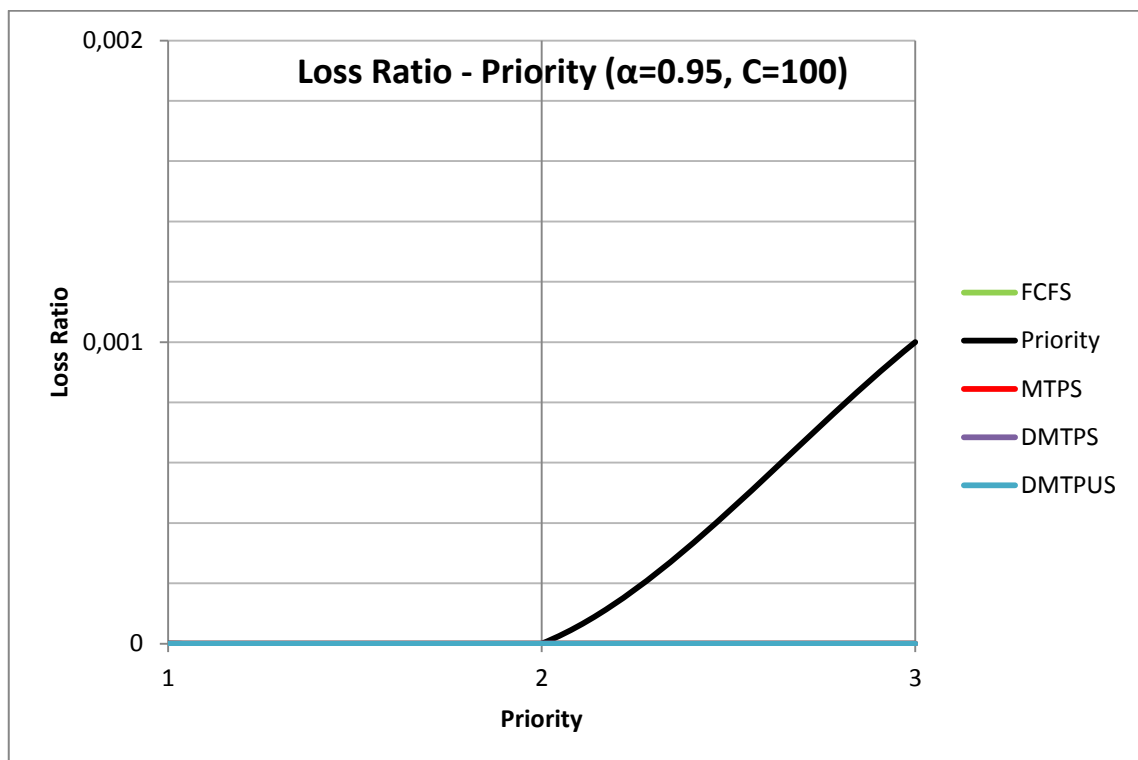


Figure 5.4 – Loss ratio - Priority graph for all algorithms, when throughput value is 0.95 and capacity is 100.

**FCFS:** Changing capacity value 50 to 100 decreases loss ratio of packets to zero for each priority level. Each priority level's loss ratio value is same.

**Priority Queue:** Changing capacity value 50 to 100 decreases loss ratio value of packets and the loss ratio of Priority 1 and Priority 2 level are zero. The lowest priority level which is numerated 3 is 0.001.

**MTPS:** According to Figure 5.3, reducing of loss ratio is observed with increasing of WQ value to 100. The values of each priority level are zero.

**DMTPS:** The value of loss ratio for each priority level is decreased according to Figure 5.3. The values of each priority level are zero same as MTPS.

**DMTPUS:** All priority level's loss ratio is decreased to zero such as FCFS, MTPS and DMTPUS.

### Capacity of waiting queue is 50 and $\alpha$ is 0.99

Capacity of waiting queue is assigned to 50. Expected service time is experimented for 9.9ms and Service rate ( $\mu$ ) is assigned to  $\frac{10}{99}$ .

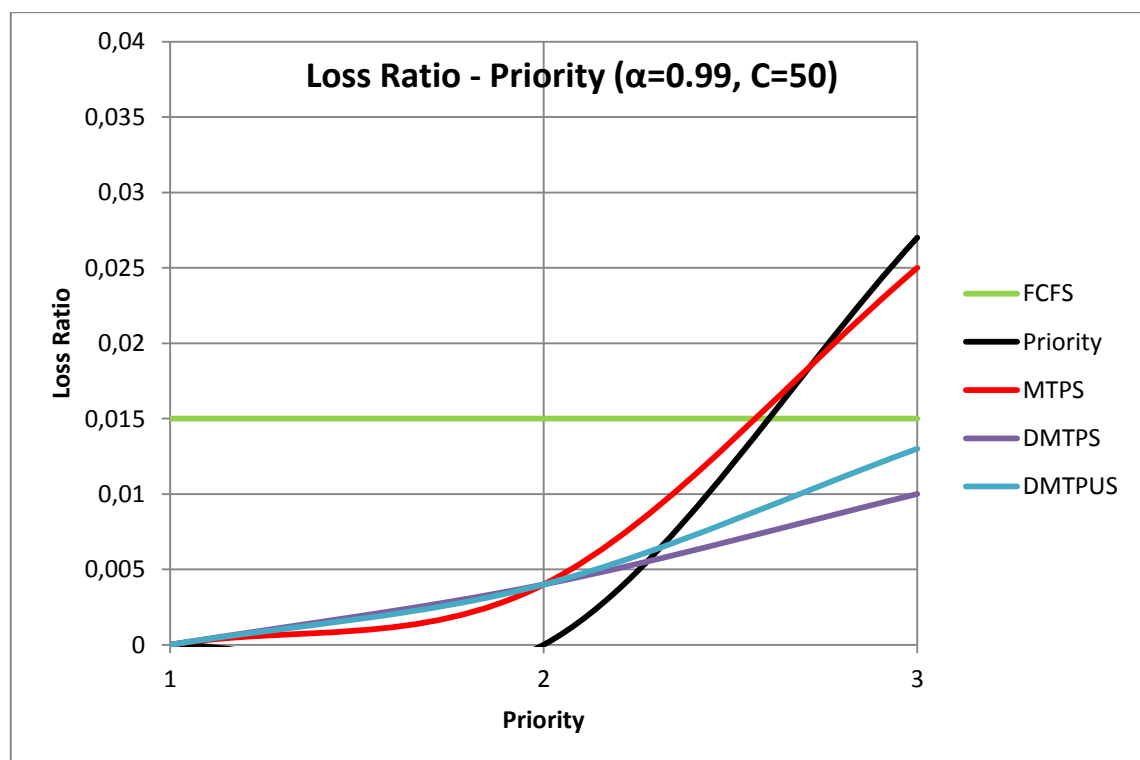


Figure 5.5 - Loss ratio - Priority graph for all algorithms, when throughput value is 0.99 and capacity is 50.

FCFS: The value of loss ratio for each priority level is same when capacity is 50 and  $\alpha$  is 0.99. The values of loss ratio are 0.015.

Priority Queue: Loss ratios of packets for each priority level are not same as FCFS. Packet which has higher level priority as numerated 1 and 2's loss ratio is 0. The lowest priority level which is numerated as 3's loss ratio is 0.025. Specially, the loss ratio of the lowest priority packets is very high.

MTPS: The logarithmic increase is observed by ascending priority levels. The higher priority packets get loss less than the lower priority packets. The highest priority packets value of loss ratio is equal to 0. The lowest priority packets value of loss ratio is equal to 0.024.

DMTPS: Loss ratio values are similar to MTPS at level of priority 1 and 2. The lowest priority packets get loss less than MTPS which value of loss ratio is 0.01.

DMTPUS: Comparison with DMTPS observes that they have similar loss ratio values for priority level 1 and 2. The most significant difference is the lowest priority packets get loss more than DMTPUS which is 0.013.

### **Capacity of waiting queue is 100 and $\alpha$ is 0.99**

Capacity of waiting queue is assigned to 100. Expected service time is experimented for 9,9ms and Service rate ( $\mu$ ) is assigned to  $\frac{10}{99}$ .

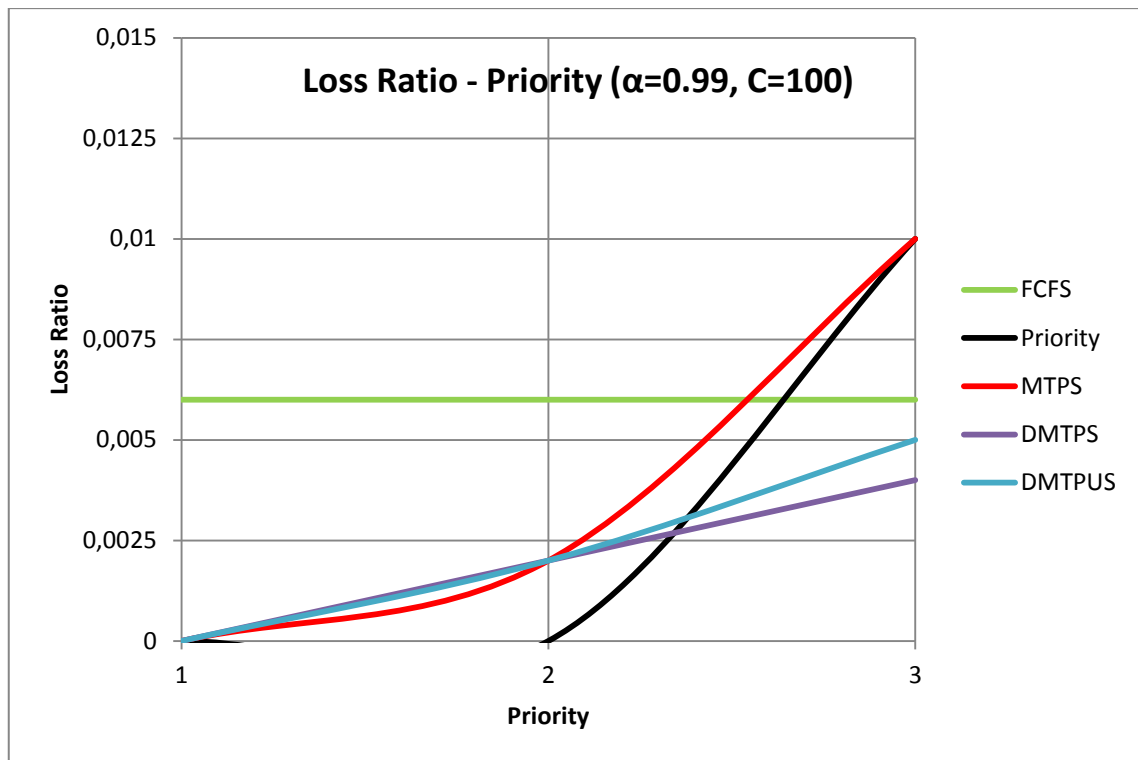


Figure 5.6 – Loss ratio - Priority graph for all algorithms, when throughput value is 0.99 and capacity is 100.

FCFS: The losses are similar for each priority level and loss ratio value of each priority level is same and it decreases from 0.015 to 0.006, according to Figure 5.5.

Priority Queue: The loss ratio value of each priority level is similar to Figure 5.4. The lowest priority packets value of loss ratio is higher than loss ratio of higher priority packets. Decreasing loss ratio values for the lower priorities are observed because of increasing capacity of Waiting Queue.

MTPS: Decreasing of loss ratio values for all priority levels is observed with increasing of capacity of waiting queue, according to Figure 5.5. The loss ratio value of lowest priority packets decreases from 0.024 to 0.01.

DMTPS: The values of loss ratio decrease because value of capacity is increased 50 to 100. MTPS and DMTPS have similar loss ratio values for priority level 1 and 2, but the lowest priority packets get loss less than MTPS. The lowest priority level loss ratio is 0.004.

DMTPUS: Packets of loss ratio which their priorities are 1 and 2 are same as MTPS and DMTPS, but the lowest priority level has lower loss ratio value than MTPS and higher loss ratio than DMTPS.

### 5.2.1 Priority-1 Level Loss Ratio Analysis

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.95

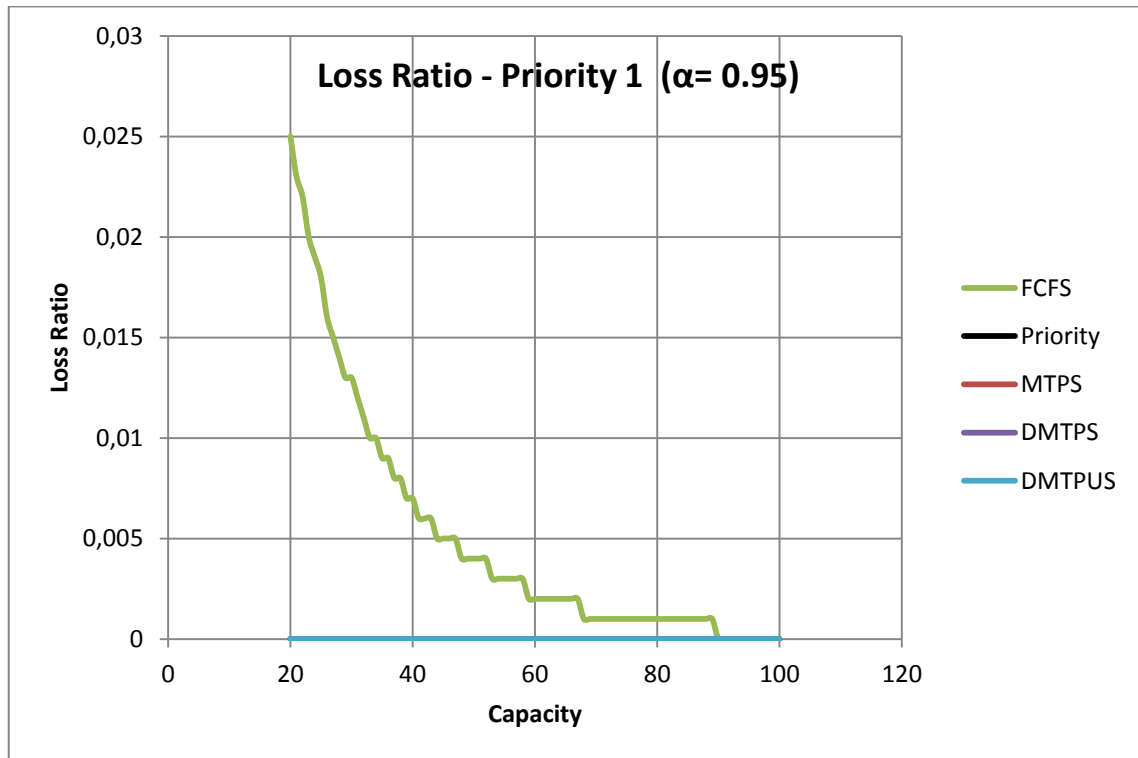


Figure 5.7 - Loss Ratio (Priority 1) - Capacity graph, when throughput value is 0.95

All scheduling algorithms have same loss ratio results (their loss ratio is equal to 0) except FCFS at Priority 1 level. The loss ratio of Priority 1 level packets at the FCFS algorithm are dropped less with increasing of capacity at Figure 5.7.

**Capacity of waiting queue is up to 100 and  $\alpha$  is 0.99**

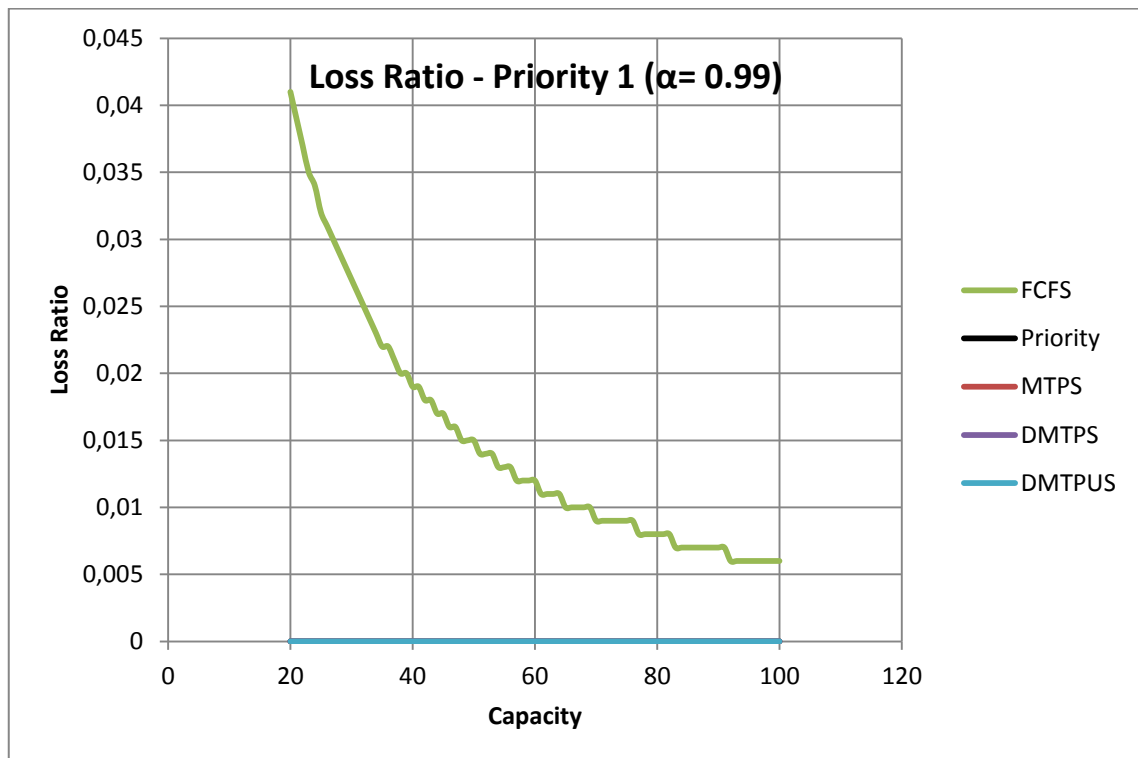


Figure 5.8 - Loss Ratio (Priority 1) - Capacity graph, when throughput value is 0.99

Result of loss ratio comparison is similar with Figure 5.7. All scheduling algorithms except FCFS have similar results at Priority 1 level. The only difference with Figure 5.7 is increasing of loss ratio for FCFS algorithm which arises from increasing of throughput value 0.95 to 0.99 at Figure 5.8.

### 5.2.2 Priority-2 Level Loss Ratio Analysis

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.95

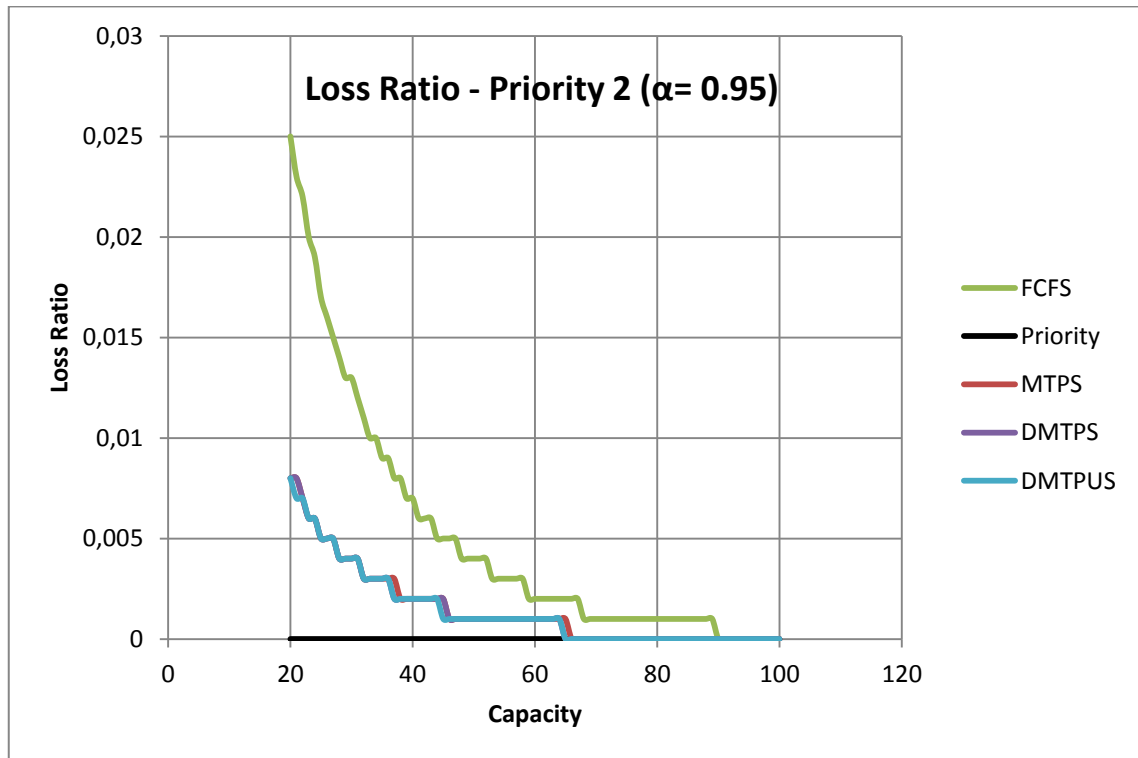


Figure 5.9 - Loss Ratio (Priority 2) - Capacity graph, when throughput value is 0.95

Figure 5.9 shows that FCFS algorithm has the highest loss ratio result and Priority Queue scheduling has the lowest loss ratio result with the changing of capacity. Threshold scheduling algorithms have similar results and their loss ratio place between FCFS and Priority Queue algorithm.



### Capacity of waiting queue is up to 100 and $\alpha$ is 0.99

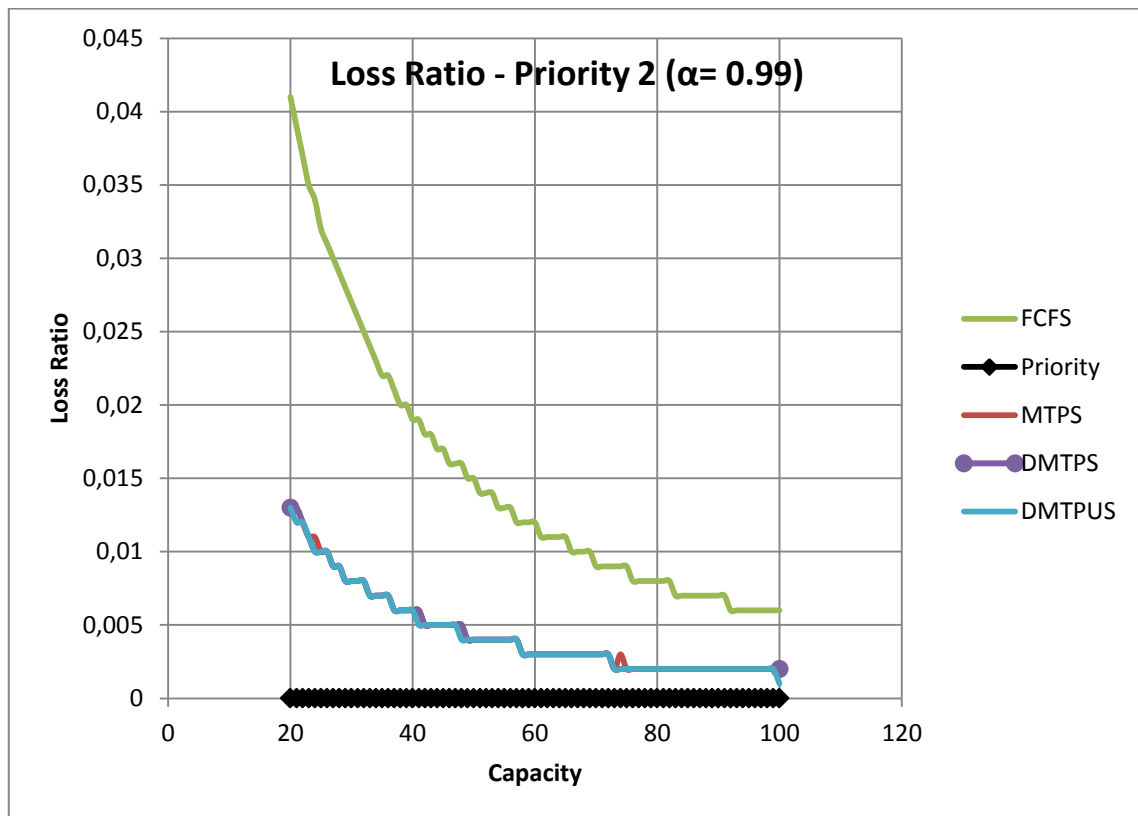


Figure 5.10 - Loss Ratio (Priority 2) - Capacity graph, when throughput value is 0.99.

Figure 5.10 shows that, there are some similarities with Figure 5.9. FCFS has the highest loss ratio values and Priority Queue has the lowest loss ratio values. Threshold scheduling algorithms have similar loss ratio values. Difference with Figure 5.9 arises from increasing of throughput value 0.95 to 0.99. It should be considered that Priority Queue scheduling algorithm has the most efficient loss ratio values for Priority-2 level. Although threshold scheduling algorithms are not as good as Priority Queue scheduling algorithm, it is notable that they have closest values of Priority Queue.

### 5.2.3 Priority-3 Level Loss Ratio Analysis

Capacity of waiting queue is 20 up to 100 and  $\alpha$  is 0.95

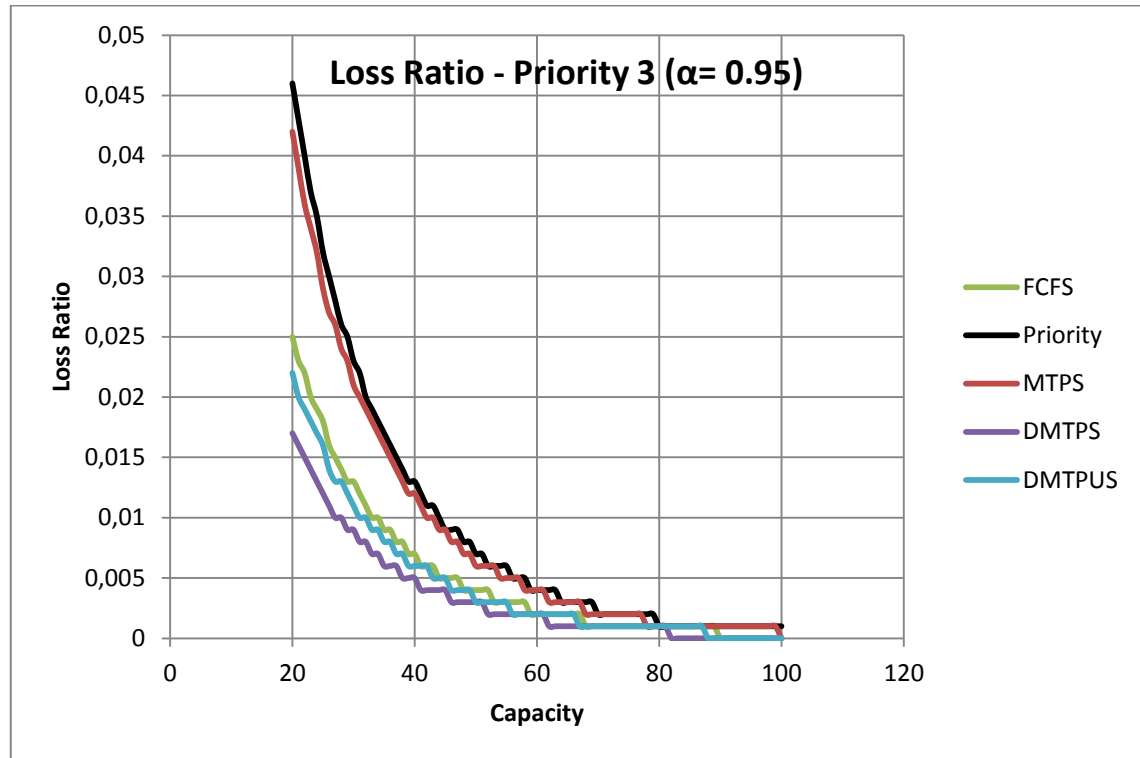


Figure 5.11 - Loss Ratio (Priority 3) - Capacity graph, when throughput value is 0.95.

Figure 5.11 shows that Priority Queue and Multi Threshold Priority scheduling algorithms have the highest loss ratio values. Specially, it is seen that dynamic threshold algorithms has loss less than the other algorithms at Priority-3 level. FCFS algorithm is also assumed as efficient for Priority-3 level packets. The most efficient algorithms for less loss ratio at Priority-3 level are dynamic threshold algorithms. As a result, Dynamic Multi Threshold Priority Scheduling algorithm has the best loss ratio values for Priority-3 level packets.

### Capacity of waiting queue is up to 100 and $\alpha$ is 0.99

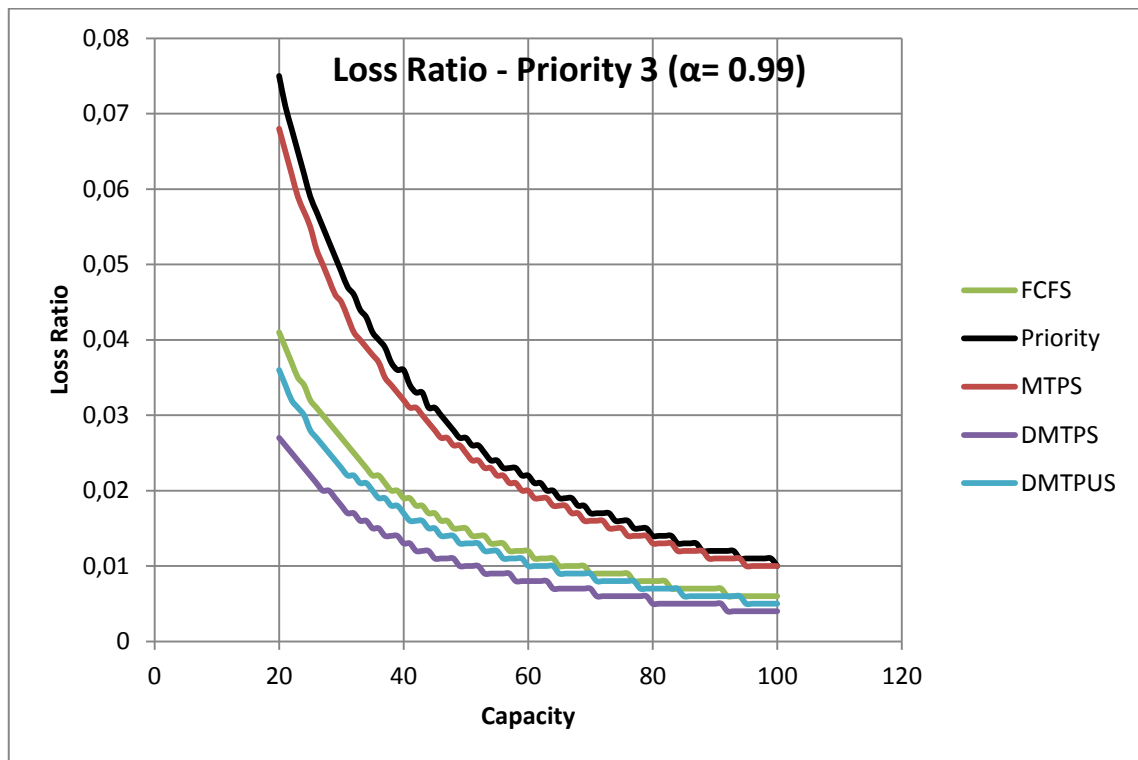


Figure 5.12 - Loss Ratio (Priority 3) - Capacity graph, when throughput value is 0.99.

Figure 5.12 show that increasing of throughput value makes evaluation of analysis clearly. Priority Queue and MTPS has the worst loss ratio values and DMTPS has the lowest loss ratio values for Priority-3 level packets. FCFS and DMTPUS algorithm are ranked as efficient algorithms for loss ratio at Priority-3 level. As a result, DMTPS is the most efficient algorithm for Priority-3 level packets and DMTPUS algorithm is not as good as DMTPS, but it should be ranked as efficient algorithm.

### 5.3 Third Phase – Total Delay Time Analysis

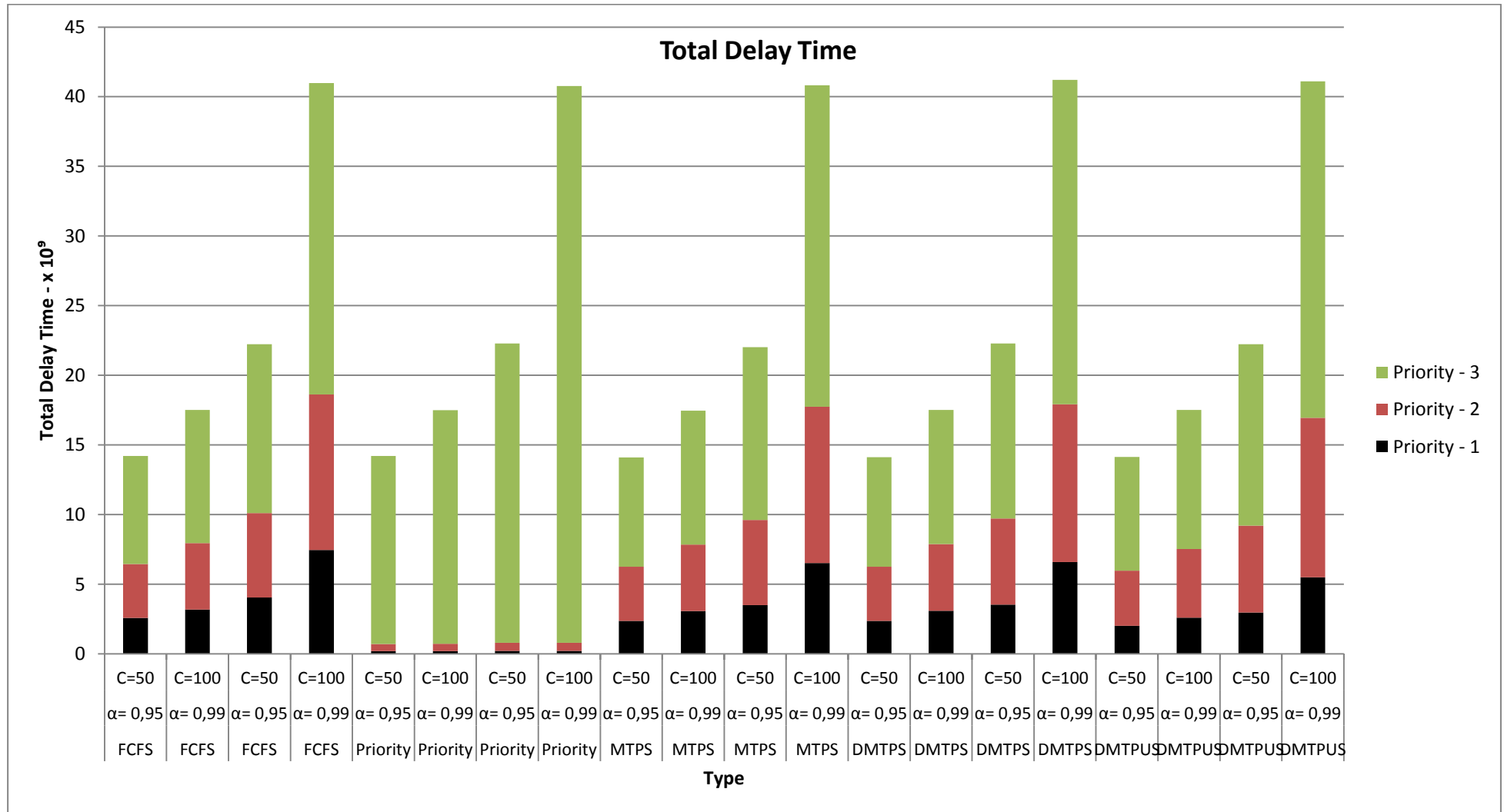


Figure 5.13 - Total Delay Time Analysis for all algorithms with different options

Total delay time analysis is produced for researching comparison of algorithms' difference reactions. Different values of capacity of Waiting Queue and service rate are experimented that make analysis better. Values of Waiting Queue capacity are 50 and 100. Expected arrival time is 10ms (millisecond) and arrival rate ( $\lambda$ ) is assigned to  $\frac{1}{10}$ . Expected arrival time is experimented for 9.5ms and 9.9ms values. Service rate ( $\mu$ ) is assigned to  $\frac{10}{95}$  and  $\frac{10}{99}$  respectively. All algorithms are run with these values. The result is graphed and seen at Figure 5.13.

The total delay time of packets are seemed similar for all algorithms. The lowest priority level (Priority-3) has more delay time with the comparison of Priority-1 and Priority-2 level. FCFS algorithm and Priority Queue algorithm have opposite delay time results except at their Priority-3 level delay time result. Priority-1 and Priority-2 level packets have the least delay time at Priority Queue algorithm. At FCFS algorithm, Priority-1 and Priority-2 delay times are higher than Priority Queue algorithm, but the lowest priority level (Priority-3) has less delay time.

Threshold scheduling algorithms have similar results with each other. Their results seem similar with FCFS algorithm. The difference with threshold algorithms and FCFS is that packets which have the highest priority (Priority-1 level) have less delay time than FCFS. MTPS algorithm has similar results with FCFS algorithm but DMTPS and DMTPUS algorithms have different results for Priority-1 level. Specially, the significant difference for Priority-1 level's delay time is seemed. The highest priority packets have less delay time at DMTPS and DMTPUS algorithm but the worst case is the lowest priority packets have higher delay time with comparison of FCFS.

As a result, higher level priority packets have less delay times but the lowest priority packets have huge delay times at Priority Queue scheduling algorithm. FCFS algorithm has advantages for the lowest priority packets' delay times but the higher priority packets have more delay times compared with Priority Queue scheduling algorithm. Threshold algorithms have similar results as FCFS. They have some improvements on the highest priority packets (Priority-1 level) delay time. DMTPS and DMTPUS scheduling algorithms provide that the

highest priority packets have less delay time compared to FCFS. These algorithms have disadvantage that the lowest priority packets (Priority-3) have more delay time results with compared to FCFS.

#### 5.4 Fourth Phase – Delay Time Ratio Per Each Priority Level

Total delay time analysis is produced for researching comparison of algorithms' difference reactions. Different values of capacity of Waiting Queue and service rate are experimented that make analysis better. Values of Waiting Queue capacity are beginning 20 to 100. Expected arrival time is 10ms (millisecond) and arrival rate ( $\lambda$ ) is assigned to  $\frac{1}{10}$ . Expected arrival time is experimented for 9.5ms and 9.9ms values. Service rate ( $\mu$ ) is assigned to  $\frac{10}{95}$  and  $\frac{10}{99}$  respectively. All algorithms are run with these values. Results are shown at Figure 5.14, 5.15, 5.16, 5.17, 5.18 and 5.19 for specific priority level.

### 5.4.1 Priority-1 Level's Delay Time Ratio Analysis

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.95

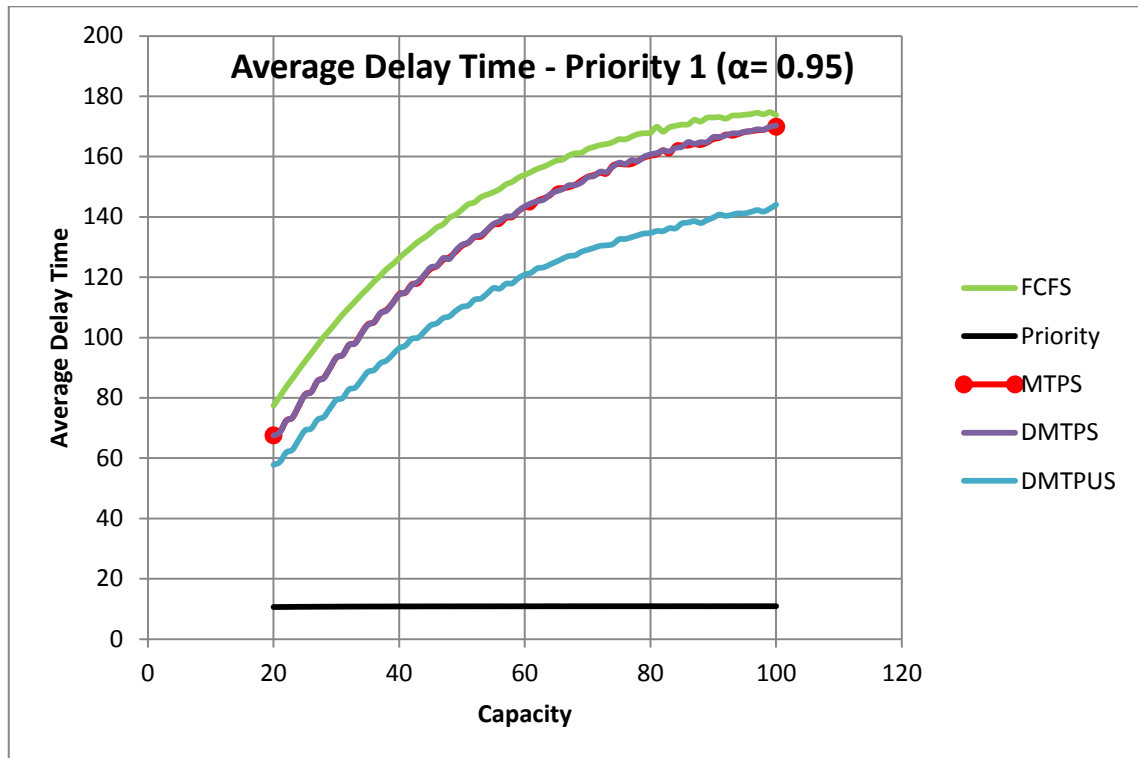


Figure 5.14 - Average Delay Time (Priority 1) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms.

Figure 5.14 shows that, Priority-1 level packets at the Priority Queue scheduling algorithms have the least average delay time. It has slightly alteration of average delay time are seemed with increasing of capacity. FCFS algorithm has the highest average delay time for the highest priority level packets and it is seemed as inefficient. MTPS and DMTPS algorithms have an advantage on average delay time and it seems they have similar results at Priority-1 level packets with compared to FCFS algorithm. DMTPUS algorithm has an ideal average delay time results and algorithm have better results according to FCFS, MTPS and DMTPS algorithms.

**Capacity of waiting queue is up to 100 and  $\alpha$  is 0.99**

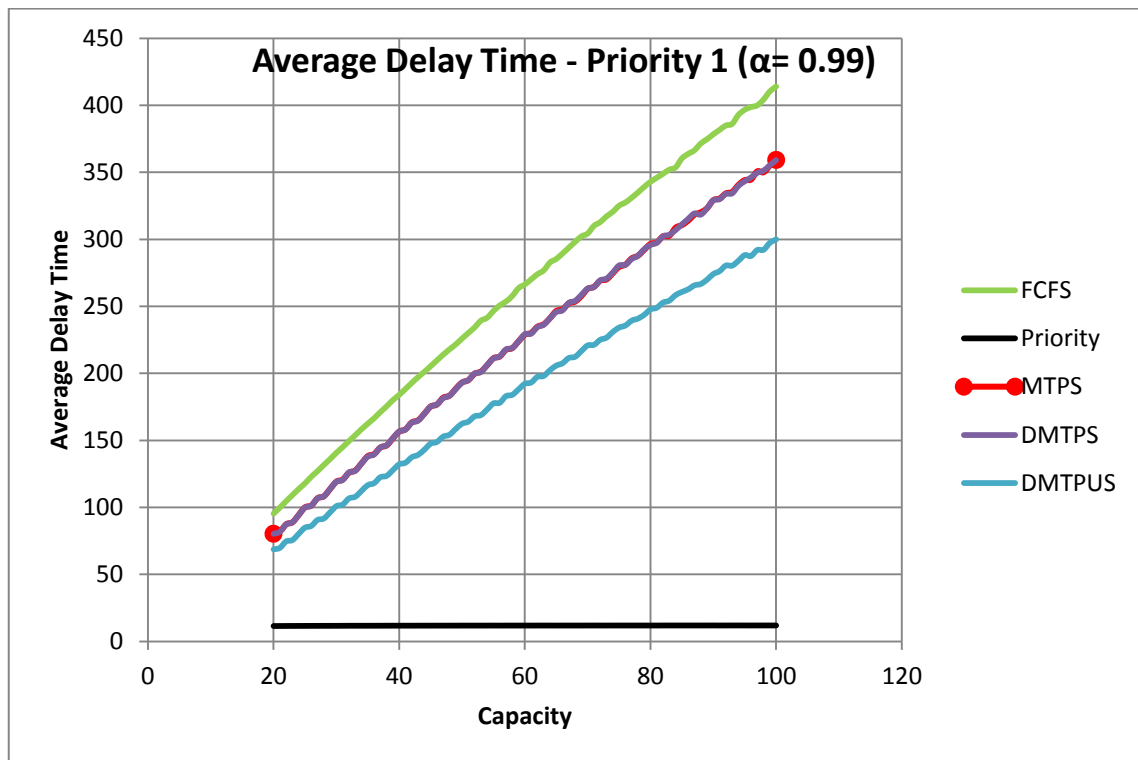


Figure 5.15 - Average Delay Time (Priority 1) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms.

Figure 5.15 shows that all algorithms except Priority Queue scheduling algorithm increases Priority-1 level packets average delay time result. At Figure 5.14, MTPS, FCFS, DMTPS and DMTPUS algorithms increase negatively, but Figure 5.15 shows that their increasing of average delay time increase positively and these increasing is advanced with the increasing of capacity. DMTPUS algorithm has an ideal average delay time results and algorithm have better results according to FCFS, MTPS and DMTPS algorithms as Figure 5.14.



### 5.4.2 Priority-2 Level's Delay Time Ratio Analysis

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.95

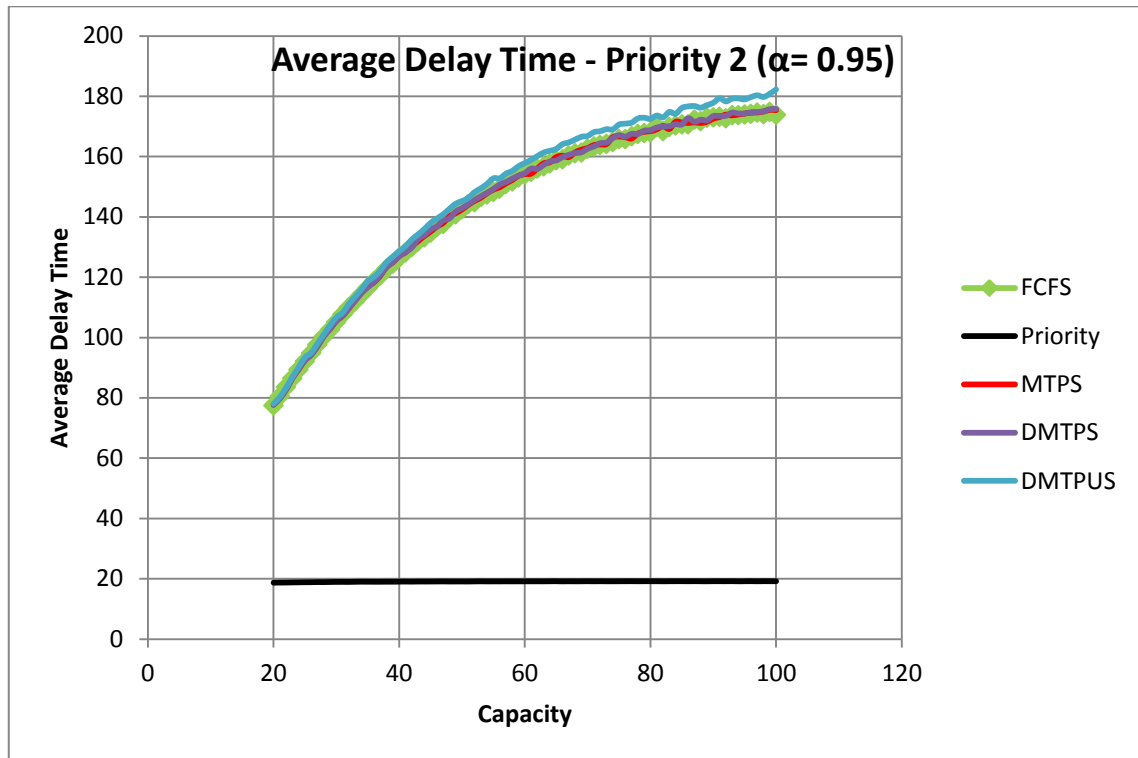


Figure 5.16 - Average Delay Time (Priority 2) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms.

Figure 5.16 shows that Priority Queue scheduling algorithm has the most advantage on average delay time for Priority-2 level packets. FCFS, MTPS, DMTPS and DMTPUS algorithms give similar results. The only difference is observed that DMTPUS algorithm has higher average delay times comparing with these algorithms. This difference is observed while capacity of Waiting Queue increases.

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.99

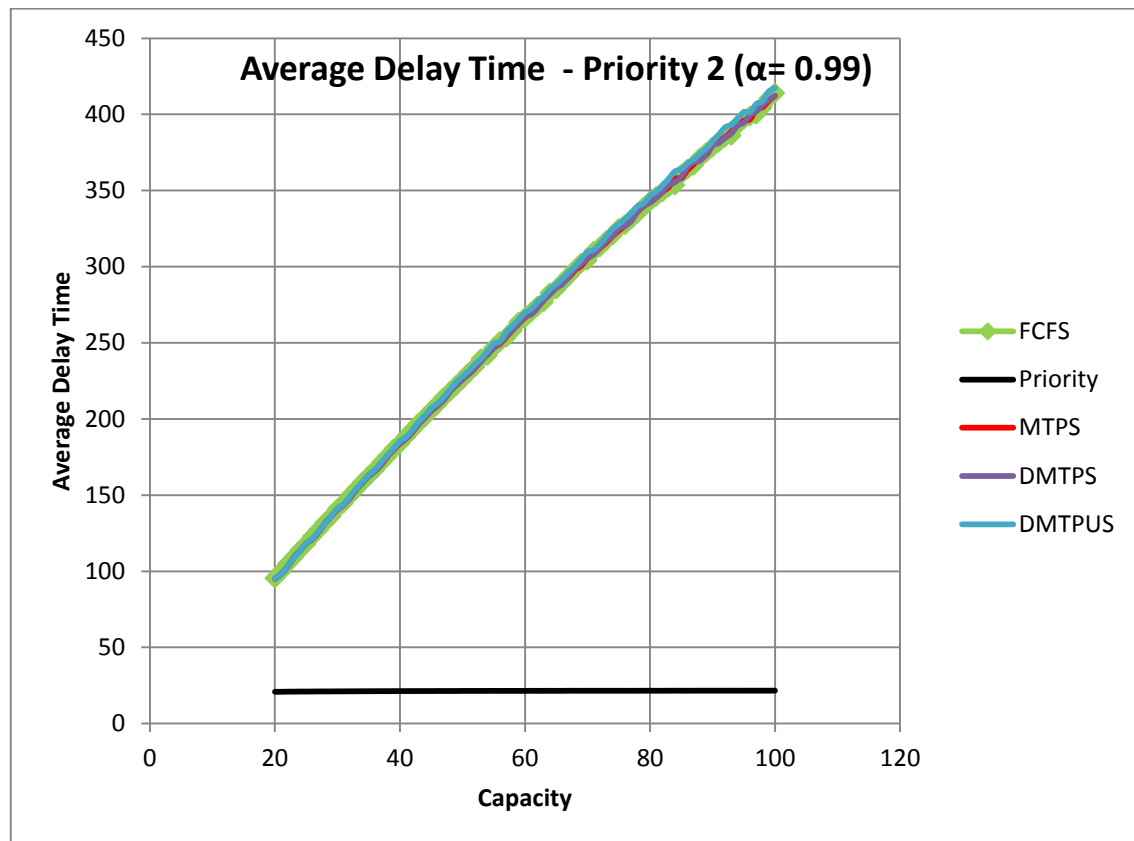


Figure 5.17 - Average Delay Time (Priority 2) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms.

Figure 5.17 shows that negative increasing on average delay times for FCFS, MTPS, DMTPS and DMTPUS algorithms' average delay time results at Figure 5.16 are replaced by positive increase with increasing of throughput value 0.95 to 0.99. Their average delay time at Priority-2 level packets increases while capacity of Waiting Queue increases. These algorithms have similar results for Priority-2 level packets' average delay time. Priority Queue scheduling algorithm has similar results as Figure 5.16.

### 5.4.3 Priority-3 Level's Delay Time Ratio Analysis

Capacity of waiting queue is up to 100 and  $\alpha$  is 0.95

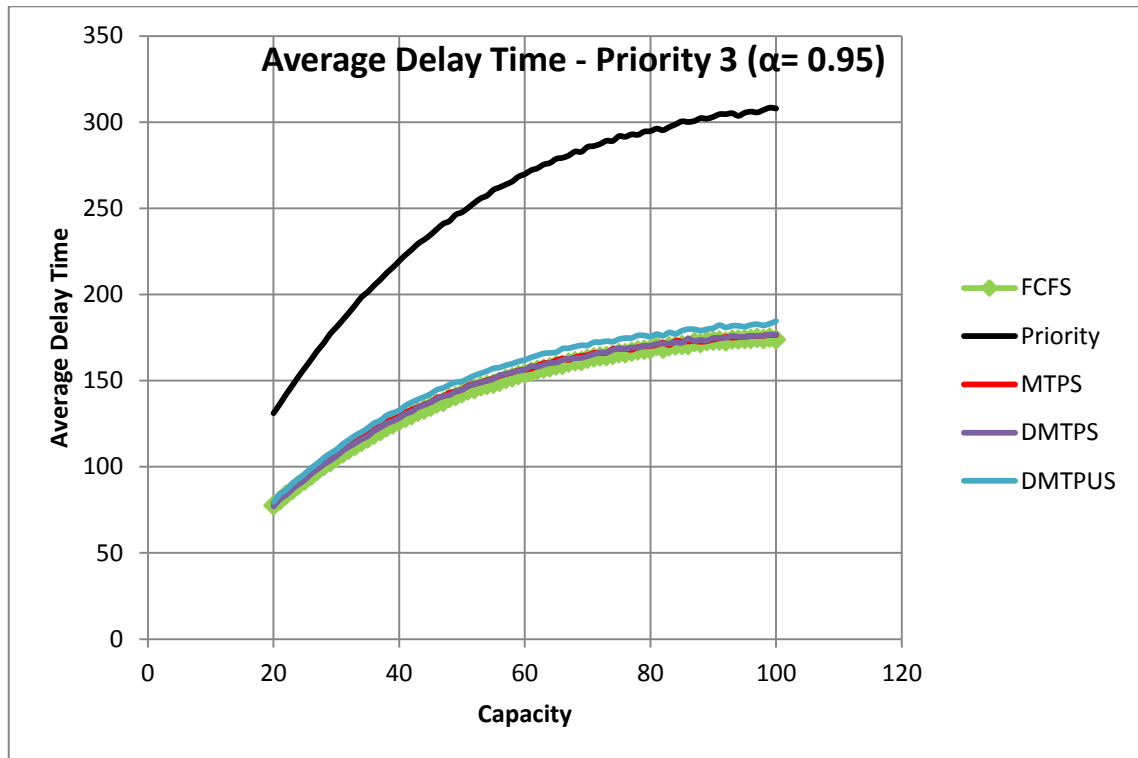


Figure 5.18 - Average Delay Time (Priority 3) – Capacity ( $\alpha = 0.95$ ) graph for all algorithms.

Figure 5.18 shows that Priority Queue scheduling algorithm is the most inefficient algorithm for Priority-3 level packets. This algorithm has the highest average delay time results. Other algorithms have similar results on average delay time at Priority-3 level packets. The only difference is observed with increasing capacity of Waiting Queue, DMTPUS have higher average delay times.

**Capacity of waiting queue is up to 100 and  $\alpha$  is 0.99**

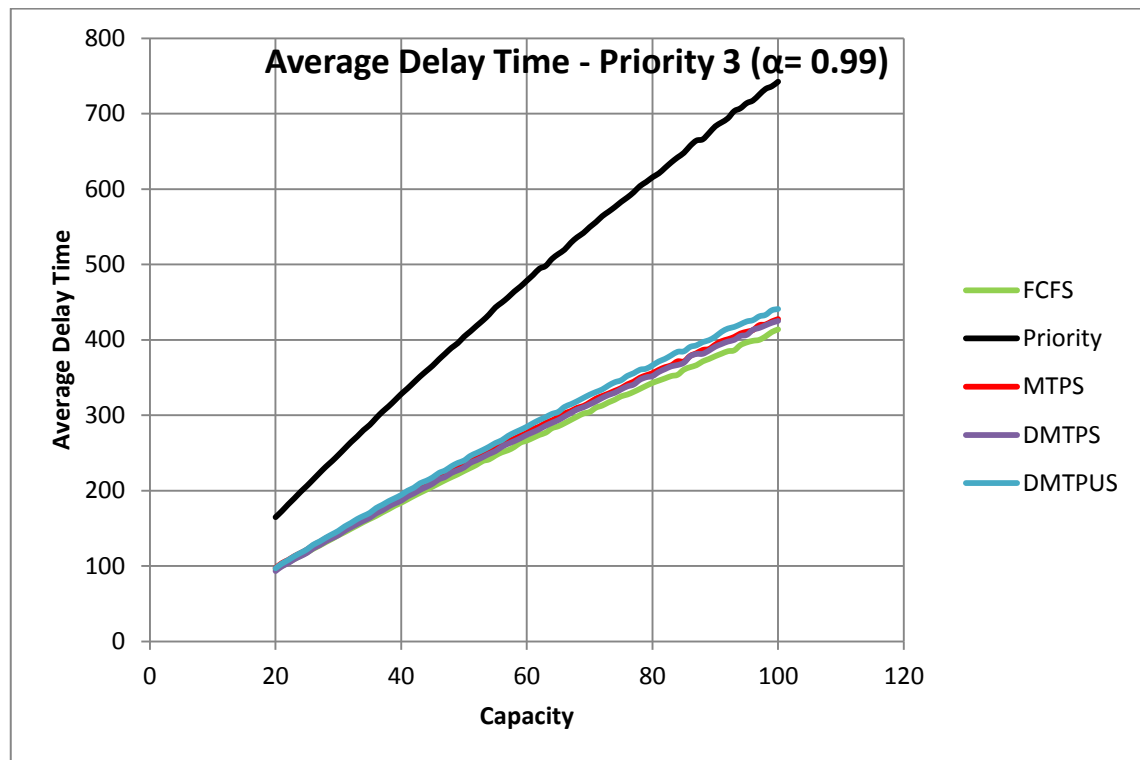


Figure 5.19 - Average Delay Time (Priority 3) – Capacity ( $\alpha = 0.99$ ) graph for all algorithms.

Figure 5.19 shows that similar results are observed as Figure 5.18. The only difference with Figure 5.18 is that negative increases are replaced by positive increases. The most efficient algorithm is FCFS, but MTPS, DMPTS and DMPTUS have good results on average delay time for Priority-3 level packets.

The result of Total Delay Time and Average Delay Time Analysis, all scheduling algorithms have same total delay time and operation of scheduling algorithms are shaped own average delay time analysis for each priority level. FCFS and Priority Queue scheduling algorithm have opposite results on delay times and Threshold algorithms have different explanation on delay time analysis. As a result, the efficient algorithms are observed as DMTPS and DMTPUS which are efficient at higher priority level packets delay time results.

## 5.5 Total Evaluation

The throughput value of a system is a factor that has impact on value of loss ratio and delay time results. The value of loss ratio increases with increasing of  $\alpha$  value. Capacity of Waiting Queue is a factor - such as  $\alpha$  -that it has very high impact on value of loss ratio. The total loss ratio value decreases with increasing of capacity.

First-Come-First-Serve, Fixed Priority Non Pre-Emptive Priority Scheduling as Priority Queuing, MTPS, DMTPS and DMTPUS algorithms indicate similar responds when the throughput value of system gets lower value. When  $\alpha$  is equal to 0.95, the value of loss ratio for each priority level is assumed as same (0).

Some alteration at value of loss ratio of each priority level is observed with increasing of value of  $\alpha$ . When  $\alpha$  is increased to 0.99 for First-Come-First-Serve algorithm, all loss ratio value of priority levels are similar to each other. Because, priority value of packet is not important for First-Come-First-Serve algorithm. Total delay time and average delay times are similar and equal to each other for all priority levels. In the Priority Queuing algorithm, the higher priority packets which priorities are 1 and 2 level's loss ratio are less than packets which have the lowest priority level. Because priority is important at the Priority Queue and order is specified according to packet's priority. In Priority Queue algorithm, the lowest priority packets loss highly. Total delay time level and average delay time of each priority is associated with loss ratio results. Threshold algorithms are assumed as higher Quality of Service for the lowest priority according to Priority scheduling.

MTPS, DMTPS and DMTPUS algorithms have similar loss ratio results. Loss ratio results of packets which are higher priority level (Priority-1 and Priority-2) are similar with Priority Queue scheduling algorithm and the lowest priority level packets loss ratio results are similar with FCFS algorithm. Threshold algorithms get new comment to loss ratio results and these algorithms provide efficiency to all priority levels. The significant alteration is observed at the lowest priority level packets loss ratio results.

Delay time results of threshold algorithms have similar results. Comparing threshold algorithms with Priority Queue and FCFS algorithms have different results at average delay time analysis. The lowest priority level packets are delayed more than threshold algorithms,

but the higher priority level packets are delayed less than threshold algorithms. FCFS has consistent results on all priority levels. Threshold algorithms aim at the lowest priority packets delay less than Priority Queue scheduling algorithm and the higher priority packets delay less than FCFS algorithm. The efficiency on the average delay time is reached with dynamic threshold algorithms. DMTPS and DMTPUS algorithms provide improvement on all priority level's loss ratio and delay time results.

To sum up, the highest and the lowest priority packets loss ratio results are seen as improvable to change. The effective way for decreasing of lower priority packets loss ratio is to produce alternative algorithms and use efficiently for all priority levels. The main purpose of threshold algorithms' creation is to minimize lower priority packets loss ratio results also with consistent delay times according to other algorithms with increasing of Quality of Service (QoS). Finally, dynamic threshold algorithms are assumed as efficient algorithms and improve Quality of Service.

## 6 CONCLUSIONS AND FUTURE WORK

In this chapter, the thesis work and its contribution are presented. All algorithms are experimented and analyzed for improving various packet scheduling algorithms' QoS.

First-Come-First-Serve (Chapter 3.2.1) and Fixed Priority Non Pre-Emptive (Chapter 3.2.2) scheduling algorithms are analyzed. Three priority levels are created and level of priority is specified to packets. The irrelevant results are obtained by comparison of priority levels. Loss ratio and delay time of packets are considered as the main factors of these results. In First-Come-First-Serve algorithm, same loss ratio is obtained per each priority level. Each priority level has same loss ratio and it is same for each levels. In Fixed Priority Non Pre-Emptive scheduling algorithm, loss ratio of each priority level is obtained as different from First-Come-First-Serve algorithm. Higher priority packets have very low loss ratio and lower priority levels have very high loss ratio results. It is obtained that there is a massive distinction between higher and lower priority levels. Irrelevant results are obtained with FCFS and Fixed Priority Non Pre-Emptive scheduling algorithm. According to these irrelevant results, the thesis work shows up.

Multi Threshold Priority Scheduling, Dynamic Multi Threshold Priority Scheduling and Dynamic Multi Threshold Priority with Urgent Scheduling are designed. Difference of Fixed Non Pre-Emptive Threshold algorithm and these algorithms are that there is not only one threshold value. The other difference with Fixed Priority Non Pre-Emptive Threshold scheduling algorithms and DMTPUS algorithm is that packets have urgency status which represents quick response at DMTPUS algorithm. Urgency status is created for elimination of unnecessary packets. In these algorithms, there are multi threshold and it is assigned for each priority level. Also, thresholds are dynamic and it can be changed periodically according to priority level. The main aim of these algorithms is standardize loss ratio per priority levels and increase Quality of Service for lower priority packets.

All of threshold algorithms aim that changing lower priority packets with higher priority packets according to own threshold value. Changed packets are placed more appropriate place to improve systems efficiency. In MTPS algorithm, all priority level has one threshold level. According to loss ratio results of MTPS algorithm, it is clearly seen as an ideal algorithm than

FCFS algorithm for higher priority level packets, but worse results for lower priority packets are obtained. Average delay time result of MTPS algorithm is similar-even slightly better - with FCFS algorithm for higher priority level packets. Comparison MTPS algorithm with Fixed Priority Non Pre-Emptive scheduling algorithm has similar-even slightly better -loss results for higher priority level. Average delay time result of MTPS algorithm has worse results for higher priority packets than Fixed Priority Non Pre-Emptive scheduling algorithms, but good loss ratio results for lower priority packet's is obtained.

DMTPS algorithm has a dynamic threshold system. Each priority level has an own threshold level. If higher priority packets do not find a place to get in to queue on its own threshold level, can be replaced a lower threshold level if exist. Aim of dynamic threshold system is to protect higher priority data in system and maintain system stability. DMTPS has a good result at loss ratio and average delay time results compared to MTPS. Better loss ratio result for lower priority packets and better average delay time result for higher priority packets has been showed clearly. DMTPS is better and more efficient algorithm than MTPS terms of loss ratio and delay time results.

DMTPUS has a dynamic threshold system such as DMTPS algorithm. The only difference with DMTPS algorithm is to quick response which has urgency status of packets is aimed. Comparison results with DMTPS shows that lower priority packets' loss ratio and average delay time results are worse than DMTPS. The only good side of DMTPUS in this comparison is that higher priority packets have better average delay time because of urgency factor. DMTPUS has better QoS than MTPS because of dynamic threshold structure, but DMTPS is assumed as has best QoS results.

In this paper, we propose Dynamic Multi Threshold Priority packet scheduling algorithms for WSNs. These algorithms ensure to decrease in delay time and loss ratio for the lower priority levels with acceptable fairness towards higher priority data. Dynamic Multi Threshold Priority packet scheduling algorithms have better performance than the existing FCFS and fixed priority non-preemptive scheduling algorithm in terms of the average delay time and loss ratio. All algorithms are analyzed according to contribution of QoS. The results show that proposed algorithms improve the QoS of selected attributes. DMTPS algorithm provides higher QoS than DMTPUS algorithm which algorithm also has a good QoS in decreasing of average delay time. Urgency and threshold optimization factors are another



factor that can be improved for future works. Finally, DMTPS algorithm is selected as the most effective algorithm for our research study.

## 7 REFERENCES

- [1] Ian F. Akyildiz, W. Su, Y.Sankarasubramaniam, E. Cayirci, A Survey on Sensor Networks, Georgia Institute of Technology, IEEE Communications Magazine, August 2002
- [2] M Bernard T. and Fouchal H., A low energy consumption MAC protocol for WSN, IEEE communication conference, ICC 2012
- [3] S.Kim, Y. Kim, An energy efficient priority-based MAC protocol for wireless sensor networks, International Conference ICT Convergence (ICTC), 2012.
- [4] K. Ramamritham, J.A. Stankovic, Scheduling Algorithms and Operating Systems Support for Real-Time Systems, University of Massachusetts, Amherst, 1994
- [5] A. Iqbal, A. Zafar, B. Siddique, Dynamic Queue Deadline First Scheduling Algorithm for Soft Real Time System, IEEE International Conference on Emerging Technologies, 2005
- [6] Aloysius K. Mok, Wing-Chi Poon, Non-Preemptive Robustness under Reduced System Load, Proceedings of the 26th IEEE International Real-Time Systems Symposium, 2005
- [7] L. XinYan, L. PingPing, Research and Improvement of Real-Time Queue Scheduling Algorithm, International Forum on Information Technology and Applications, 2010
- [8] C. Taddia, G. Mazzini, On the Jitter Performance of FIFO and Priority Queues mixture, The 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, University of Ferrara, Italy, 2006
- [9] N. Nasser, L. Karim, T. Taleb, Dynamic Multilevel Priority Packet Scheduling Scheme for Wireless Sensor Network, IEEE Transactions on Wireless Communications, Vol.12, No.4, 2013

- [10] J. Banks, J. Carson, B. Nelson, D. Nicol, *Discrete-Event System Simulation, 5th Ed.*, New Jersey, 2010
- [11] J. Gosling, B. Joy, G. Steele; G. Bracha, A. Buckley, *The Java® Language Specification*, Java SE 8 ed., 2014
- [12] P. Moulin, *A note on Exponential Distribution*, 2007
- [13] W. Lamie, *Preemption-threshold*, [online] Available:  
<http://www.threadx.com/preemption.html>
- [14] Y. Wang, M. Saksena, *Scheduling Fixed-Priority Tasks with Preemption Threshold*, Hong Kong, 1999
- [15] M.L. Pinedo, *Scheduling: theory, algorithms, and systems, 4th edition*, New York, 2012.
- [16] R. R. Guadaña , M. R. Perez, L. Rutaquio Jr, *A Comprehensive Review for Central Processing Unit Scheduling Algorithm*, IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No 2, January 2013
- [17] M. Shoaib, M. Z. Farooqui, *A Comparative Review of CPU Scheduling Algorithms*, Proceedings of National Conference on Recent Trends in Parallel Computing (RTPC - 2014), November 1-2, 2014, Mangalayatan University, Aligarh, India.
- [18] V. Chahar, S. Raheja, *A Review of Multilevel Queue and Multilevel Feedback Queue Scheduling Techniques*, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 1, January 2013
- [19] J.A. Sokolowski, C.M. Banks, *Principles of Modeling and Simulation*, p. 6, ISBN 978-0-470-28943-3, 2009

## **8 CIRRICULUM VITAE**

Sezer Uzungenç was born on 8 August 1988, in Hatay. He graduated from Makzume Anatolian High School in 2006. Then, he enrolled into Kadir Has University in 2008 and graduated in 2012. He has a bachelor degree of Computer Engineering. Afterwards, he continued on his education in Kadir Has University at the graduate program. His main research interests are programming languages and designing.