Packet Queueing and Scheduling Algorithm based on Socket Programming

Jiaru Zou, Zirui Wang | Project Github Link: https://github.com/Ziruiwang409/packet-queueing-and-scheduling-algo

1. Introduction

Packet scheduling algorithms are used in computer networks to prioritize the transmission of packets (data units) over the network. These algorithms determine the order in which packets are transmitted to optimize network performance and meet Quality of Service (QoS) requirements. There are various packet scheduling algorithms, each with its own advantages and disadvantages. In this paper, we mainly focus on the following four scheduling algorithms:

First-In-First-Out (FIFO) This algorithm simply prioritizes packets based on the order in which they arrived. It is easy to implement but does not prioritize packets based on their importance or urgency.

Weighted Fair Queuing (WFQ) This algorithm assigns weights to each flow of traffic, and then schedules packets based on their weight. This allows for more granular control over network traffic but can be more complex to implement.

Priority Queuing (PQ) This algorithm separates packets into different queues based on their priority level. High-priority packets are transmitted before low-priority packets, which can help ensure that critical traffic is not delayed.

Round Robin (RR) This algorithm cycles through each flow of traffic in a round-robin fashion, transmitting one packet from each flow before moving on to the next. This can help ensure fairness in network traffic but may not be optimized for certain types of traffic.

In general, Packet scheduling algorithms are an important part of network optimization, and different algorithms may be better suited to different types of networks and traffic.

2. Related Paper Survey

Paper 1: *Scheduling algorithms in broadband wireless networks*

In the future, wireless networks are expected to play a crucial role in global communication infrastructure. With the increasing popularity of wireless data services and the demand for multimedia applications, it is necessary to develop new technologies to provide quality of service (QoS) differentiation and guarantees in wireless networks. One of the key technical issues in achieving this is packet scheduling in wireless networks.

Based on the description of the paper, Scheduling algorithms in wireless networks determine how bandwidth is allocated and multiplexed at the packet level. Existing scheduling algorithms developed for wireline networks are not directly applicable to wireless networks due to the unique characteristics of wireless communication. These characteristics include high error rates, location-dependent and time-varying link capacity, scarce bandwidth, user mobility, and power constraints of mobile hosts. These factors make it challenging to design efficient and effective scheduling algorithms for wireless networks.

Several scheduling algorithms have been proposed to address the special challenges of wireless networks. However, none of these approaches systematically tackle the problem of packet scheduling in wireless networks. Therefore, there is a need for comprehensive research in this area.

The system model for wireless networks typically involves a cell-structured network architecture where the service area is divided into cells, each with a base station. Mobile hosts communicate with the base station within a cell, and base stations are connected via wireline networks. Scheduling algorithms at the base station handle packet transmissions between the mobile hosts and the base station, considering the varying wireless link capacity and location-dependent channel states.

The schedulers are broadly classified as centralized and distributed schedulers. The distributed schedulers are suitable for ad hoc networks. The Schedulers can also be classified as work-conserving or non-work-conserving. A work-conserving scheduler is never idle if there is a packet awaiting transmission. Examples include Generalized

Processor Sharing (GPS), packet-by-packet GPS also known as Weighted Fair Queuing (WFQ), Virtual Clock (VC), Weighted Round-Robin (WRR), Self-Clocked Fair Queuing (SCFQ), and Deficit Round-Robin (DRR). In contrast, a nonwork-conserving scheduler may be idle even if there is a backlogged packet in the system because it may be expecting another higher-priority packet to arrive. Examples are Hierarchical Round-Robin (HRR), Stop-and-Go Queuing (SGQ), and Jitter-Earliest-Due-Date (Jitter- EDD). Non-work conserving schedulers generally have higher average packet delay than their work-conserving counterparts but may be used in applications where time jitter is more important than delay. In that, some of the schedulers are originally proposed for wireline networks. These algorithms can be used as the error-free service model in the design of wireless schedulers.

The major issues in designing wireless scheduling algorithms include:
1. Wireless Link Variability: Wireless links are subject to high variability due to error-prone transmission channels affected by interference, fading, and shadowing. The capacity of a wireless link can vary over time and depends on the physical location of mobile hosts. Scheduling algorithms need to dynamically adapt to these changes.

2. Fairness: Ensuring fairness in scheduling is more complicated in wireless networks compared to wireline networks. Deferring packet transmission on error-prone links can temporarily affect the fairness among different flows. Determining appropriate fairness definitions and objectives in a wireless environment is crucial.

3. Quality of Service (QoS): Future wireless networks will support heterogeneous classes of traffic with different QoS requirements. Scheduling algorithms need to differentiate and guarantee QoS for different traffic classes based on the service model. This may involve prioritized scheduling or per-flow-based QoS guarantees.

4. Data Throughput and Channel Utilization: Wireless networks have limited bandwidth, and efficient scheduling algorithms should aim to maximize the effective service delivered and channel utilization while minimizing unproductive transmissions on error-prone links.

5. Power Constraint and Simplicity: Mobile hosts in wireless networks are power-constrained, so scheduling algorithms should minimize the power required for control messages and computations. Additionally, scheduling algorithms should be designed for real-time multimedia traffic with stringent timing requirements.

Researchers have proposed various scheduling algorithms for wireless networks, such as Channel State Dependent Packet Scheduling (CSDPS) and its variant CSDPS CBQ (Class-Based Queueing). These algorithms aim to address some of the challenges mentioned above, but they still have limitations and trade-offs. Overall, packet scheduling in wireless networks is a complex and ongoing research area. Further exploration is needed to develop efficient and effective scheduling algorithms that can provide QoS differentiation and guarantees while considering the unique characteristics of wireless communication.

Paper 2: *Review of Packet Scheduling Algorithms in Mobile Ad Hoc Networks*
The paper describes the challenges and solutions related to scheduling algorithms in ad hoc networks, which are self-organizing wireless networks without any pre-existing wired infrastructure. In ad hoc networks, the mobility of nodes and the error-prone nature of the wireless medium pose challenges such as frequent route

changes and packet losses, which can lead to increased packet delays and decreased throughput. To address these issues, research has primarily focused on routing protocols and medium access control (MAC). However, there is a need to understand the queuing dynamics in the nodes of these networks and explore different packet scheduling algorithms to improve network performance.

The paper discusses the classification of scheduling algorithms based on various criteria. Schedulers can be classified as centralized or distributed, and work-conserving or non-work-conserving. Work-conserving schedulers ensure that the transmission medium is never idle if packets are waiting for transmission, while non-work-conserving schedulers may remain idle even if there are backlogged packets, waiting for higher-priority packets.

The text also highlights different types of scheduling algorithms, including fair scheduling, QoS-aware scheduling, priority scheduling, opportunistic packet scheduling, and message scheduling. Fair scheduling algorithms aim to provide low average and maximum delay for applications such as audio and video. QoS-aware scheduling algorithms consider the quality of service requirements of different flows in the network. Priority scheduling assigns higher priority to certain nodes or packets to achieve fairness and meet QoS parameters. Opportunistic packet scheduling protocols utilize techniques such as multicast RTS (Request-to-Send) and priority-based CTS (Clear-to-Send) to improve system throughput and mitigate head-of-line blocking. Distributed opportunistic scheduling algorithms incorporate cooperation among neighboring transmitters to optimize scheduling policies.

Additionally, there are scheduling algorithms that consider the size and number of hops of messages. For example, the Smallest Message First (SMF) algorithm schedules packets in ascending order of message size, while other algorithms assign priorities based on the number of hops a message has traversed. Overall, the text provides an overview of the different types of scheduling algorithms in ad hoc networks and their classification is based on various criteria. These algorithms play a crucial role in managing packet delays, improving throughput, and ensuring fairness and QoS in self-organizing wireless networks.

Paper 3: *"Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks."*
The paper introduces an extension to the Start-Time Fair Queuing (SFQ) algorithm to support variable rate allocation for video flows, enabling efficient utilization of network resources. The concept of rate assigned to a packet is introduced, along with the definition of the finished tag based on this rate.

Two types of deadline guarantees provided by SFQ are discussed: a delay guarantee based on the expected arrival time of a packet, and a delay-cum-throughput guarantee based on the arrival time and the departure time of the previous packet. These guarantees are independent of other flow behaviors and can be used to offer various quality of service (QoS) guarantees.

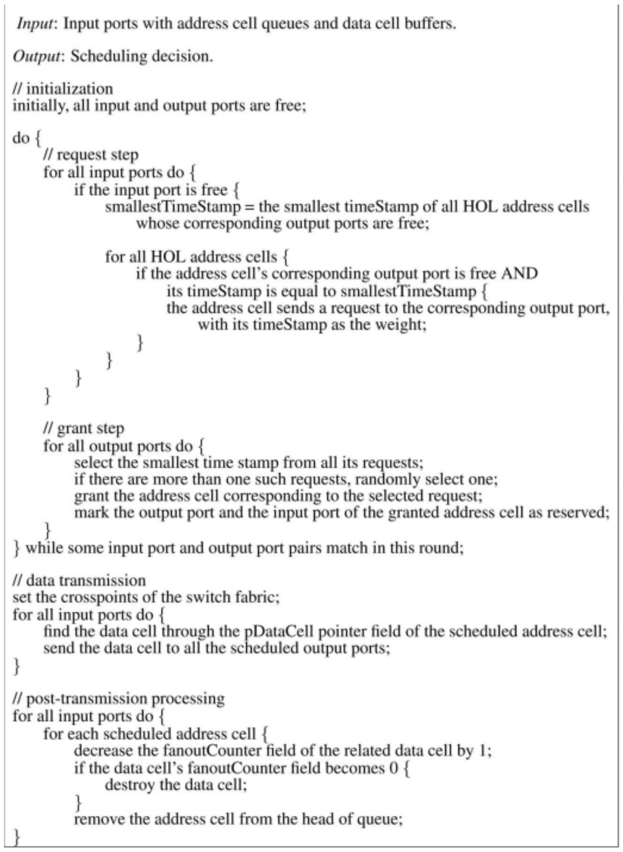
Bounds on the work done by an SFQ server within a virtual time interval are derived, and the delay guarantees for SFQ FC and EBF servers are presented. SFQ is shown to provide a lower maximum delay compared to SCFQ and WFQ, and it can reduce the average delay for low-throughput flows while increasing it for high-throughput flows.

The paper also discusses the delay-cum-throughput guarantee of SFQ servers, which improves upon the delay guarantee when the actual service received exceeds the guaranteed service. This property is particularly useful for flows with varying time-scale variations, such as video traffic.The upcoming section focuses on deriving the end-to-end delay guarantee for a network of SFQ servers and exploring its potential benefits for flow-controlled data and adaptive real-time applications.

Paper 4: *FIFO-based multicast scheduling algorithm for virtual output queued packet switches*
MULTICAST, the transmission of information from a single source to multiple destinations, is crucial for high-performance networks. Efficiently organizing and scheduling multicast packets in packet switches has been a challenging issue. There are two main types of packet switches: output-queued (OQ) switches and input-queued (IQ) switches. OQ switches achieve high throughput but face scaling difficulties, while IQ switches have been the focus for high-speed switches. However, IQ switches suffer from head-of-line (HOL) blocking, limiting their performance. Existing scheduling algorithms for IQ switches primarily target unicast traffic and struggle with multicast traffic.

To address these challenges, this paper proposes a new scheme for organizing multicast packets in a virtual output queued (VOQ) switch. The scheme separates the address information and data content of packets, reducing the number of required queues. Additionally, a first-in-first-out-based multicast scheduling algorithm called FIFO Multicast Scheduling (FIFOMS) is introduced, which efficiently utilizes the multicast capability of the switch, eliminates HOL blocking, and performs well under both multicast and unicast traffic. Simulation results demonstrate that FIFOMS outperforms other scheduling algorithms for input-queued switches in terms of average packet delay and buffer space requirements. The paper concludes with discussions on implementation issues and complexity, along with presenting the simulation results.

The algorithm inside the paper can be described below in Figure 1.



```
Input: Input ports with address cell queues and data cell buffers.

Output: Scheduling decision.

// initialization
initially, all input and output ports are free;

do {
    // request step
    for all input ports do {
        if the input port is free {
            smallestTimeStamp = the smallest timeStamp of all HOL address cells
                whose corresponding output ports are free;

            for all HOL address cells {
                if the address cell's corresponding output port is free AND
                    its timeStamp is equal to smallestTimeStamp {
                    the address cell sends a request to the corresponding output port,
                        with its timeStamp as the weight;
                }
            }
        }
    }

    // grant step
    for all output ports do {
        select the smallest time stamp from all its requests;
        if there are more than one such requests, randomly select one;
        grant the address cell corresponding to the selected request;
        mark the output port and the input port of the granted address cell as reserved;
    }
} while some input port and output port pairs match in this round;

// data transmission
set the crosspoints of the switch fabric;
for all input ports do {
    find the data cell through the pDataCell pointer field of the scheduled address cell;
    send the data cell to all the scheduled output ports;
}

// post-transmission processing
for all input ports do {
    for each scheduled address cell {
        decrease the fanoutCounter field of the related data cell by 1;
        if the data cell's fanoutCounter field becomes 0 {
            destroy the data cell;
        }
        remove the address cell from the head of queue;
}
```

Figure 1

3. Project description

Packet queuing and scheduling algorithms are essential components of modern network communication systems. They are responsible for managing the flow of packets through the network and ensuring that traffic is delivered efficiently and reliably. Bad behavior of routing mechanisms might cause serious network congestion issues. The purpose of a packet queuing and scheduling algorithm is to determine which packets should be transmitted and in what order.

In essence, these algorithms prioritize traffic based on various factors such as packet size, type of traffic, and network congestion. There are many different packet queuing and scheduling algorithms that can be used depending on the specific requirements of the network. Some examples include First-In-First-Out

(FIFO), Weighted Fair Queuing (WFQ), and Priority Queuing. In this project, we will explore different packet queuing and scheduling algorithms and evaluate their performance in different network scenarios.

4. Experiments:

We will use simulation tools such as OMNeT++ to test the algorithms and compare their efficiency, latency, and throughput. The simulator assumes four sources S1, S2, and S3 which transmit at a specific rate, and send a given number of packets (see table below) at different intervals via different paths/queues. R extracts the payloads and forwards to destination (D) using the above mentioned scheduling algorithms (figure 2) via a single path/queue. We've used UDP sockets for the IPC (Figure 3).

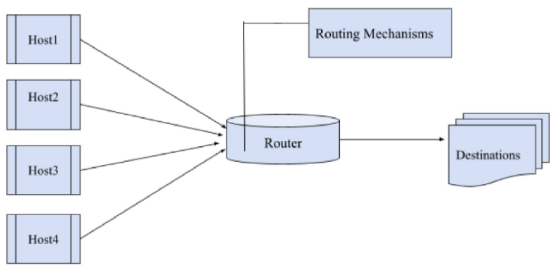


Figure 2. Scheduling Algorithms

	Packet Count	Packet Interval（ ms)	Packet Size (Bytes)
S1	100	200	100
S2	1000	30	50
S3	500	300	100

Figure 3. Packet Information

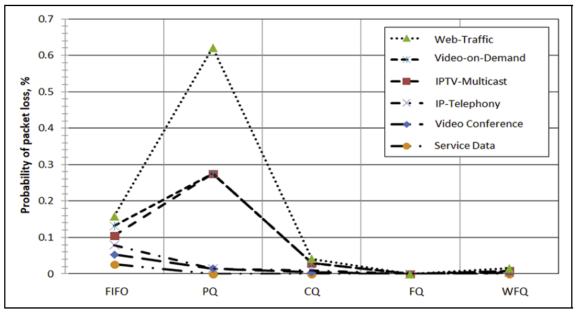
Github link for implementation:
https://github.com/Ziruiwang409/packet-queueing-and-scheduling-algo

In the experiments, we assume the weights assigned to the packets from the three different sources S1, S2, and S3. Each source sends a payload, with the first byte indicating the source number i.e. "1ksdfsdfsdfu" which implies that the packet arrived from S1. (figure 4).

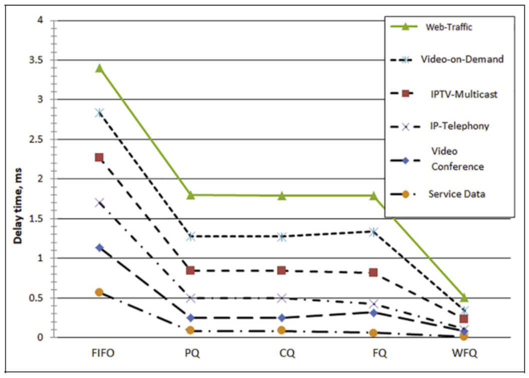
Source	Weight
S1	1
S2	2
S3	0.5

Figure 4. Sending payload of each source

The probability of packet loss for different flows using the corresponding service algorithm:



The delay time for different flows using the corresponding service algorithm.



5. Conclusion

In conclusion, the three routing algorithms - FIFO (First-In-First-Out), Round Robin, and WFQ (Weighted Fair Queuing) - are all widely used in network routing to optimize traffic flow and resource allocation. Each algorithm has its own advantages and considerations, making them suitable for different network environments and requirements.

1. FIFO (First-In-First-Out): FIFO is a simple and intuitive routing algorithm that processes network traffic in the order it arrives. It ensures fairness by treating all packets equally and is easy to implement. However, FIFO does not consider the priority or importance of packets, which can lead to potential issues such as delay or congestion for time-sensitive or high-priority traffic.

2. Round Robin: The Round Robin algorithm distributes network traffic evenly across available resources. It aims to provide equal opportunity for packet transmission and prevents any single flow from monopolizing the system. Round Robin is effective for load balancing in scenarios where all traffic has similar importance. However, it may not be suitable for environments with varying packet sizes or when some flows require different levels of service.

3. WFQ (Weighted Fair Queuing): WFQ is a sophisticated routing algorithm that assigns weights to different flows based on their priorities. It ensures fairness by considering the importance or urgency of packets and dynamically allocates bandwidth accordingly. WFQ can prioritize critical traffic and maintain quality of service for different flows. However, WFQ requires more computational resources for packet classification and can introduce additional latency due to its complexity.

The choice of routing algorithm depends on the specific needs of the network. FIFO is a straightforward approach suitable for environments where fairness is the primary concern. Round Robin is effective for load balancing and achieving resource equity among flows. WFQ provides more granular control and prioritization capabilities, making it suitable for networks with diverse traffic requirements. Network administrators should carefully assess the characteristics of their network and select the most appropriate algorithm to optimize performance and meet their specific objectives.

6. Reference

1. Yaxin Cao and V. O. K. Li, "Scheduling algorithms in broadband wireless networks," in Proceedings of the IEEE, vol. 89, no. 1, pp. 76-87, Jan. 2001, doi: 10.1109/5.904507.

2. Review of Packet Scheduling Algorithms in Mobile Ad Hoc Networks C.Annadurai Assistant Professor/ECE, SSN College of Engineering, Kalavakkam – 603 110, Chennai. Tamilnadu, India

3. P. Goyal, H. M. Vin, and Haichen Cheng, "Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks," in IEEE/ACM Transactions on Networking, vol. 5, no. 5, pp. 690-704, Oct. 1997, doi: 10.1109/90.649569.

4. D. Pan and Y. Yang, "FIFO-based multicast scheduling algorithm for virtual output queued packet switches," in IEEE Transactions on Computers, vol. 54, no. 10, pp. 1283-1297, Oct. 2005, doi: 10.1109/TC.2005.164.

5. Introduction to Cisco's Quality-of-Service Architecture for IP Networks. Vinod Joseph, Brett Chapman, in Deploying QoS for Cisco IP and Next Generation Networks, 2009

6. A. Demers, S. Keshav, and S. Shenker, Analysis and Simulation of a Fair Queuing Algorithm, *Internetworking: Research and Experience*, Vol. 1, No. 1, pp. 3-26, 1990