AGE-AWARE SCHEDULING IN COMMUNICATION NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HASAN BURHAN BEYTUR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2020

Approval of the thesis:

AGE-AWARE SCHEDULING IN COMMUNICATION NETWORKS

submitted by **HASAN BURHAN BEYTUR** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Elif Uysal
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Nail Akar
Electrical and Electronics Engineering, Bilkent University _____

Prof. Dr. Elif Uysal
Electrical and Electronics Engineering, METU _____

Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering, METU _____

Prof. Dr. Ali Özgür Yılmaz
Electrical and Electronics Engineering, METU _____

Assist. Prof. Dr. Mustafa Mert Ankaralı
Electrical and Electronics Engineering, METU _____

Date: 17.09.2020

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Hasan Burhan Beytur

Signature :

ABSTRACT

AGE-AWARE SCHEDULING IN COMMUNICATION NETWORKS

Beytur, Hasan Burhan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Elif Uysal

September 2020, 87 pages

In next-generation communication networks, various types of data traffic with different requirements will coexist. This type of coexistence requires new techniques based on new metrics to distinguish the various types of data traffic based on their value. The emerging metric Age of Information (AoI) quantifying the timeliness of the communication flow is a promising metric to prepare future networks for new technologies such as the Internet of Things and Autonomous Driving. In this thesis, we studied scheduling problems with the objective of minimizing AoI on different network models. With the help of the results obtained from the theoretical study of LCFS queues using the Stochastic Hybrid System method, we propose a prominent scheduling algorithm called Maximum Age Difference, which is useful for optimizing the network in terms of AoI. Additionally, we approached the age-minimizing scheduling problem with Reinforcement Learning-based methods, which is advantageous when the network statistics are unknown.

Keywords: Age of Information, Communication Networks, Scheduling Algorithms

ÖZ

HABERLEŞME AĞLARINDA BİLGİ YAŞI FARKINDA ÇİZELGELEME

Beytur, Hasan Burhan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Elif Uysal

Eylül 2020 , 87 sayfa

Yeni nesil haberleşme ağlarında farklı isterlere sahip olan çeşitli veri trafikleri birlikte bulunacak. Bu birliktelik yeni metriklere dayanan ve veri trafiklerini ayırt edebilen yeni tekniklere ihtiyaç duymaktadır. Haberleşme bağlantılarının tazeliğini ölçen Bilgi Yaşı metriği, yeni nesil haberleşme ağlarını Nesnelerin İnterneti ve Otonom Sürüş gibi yeni teknolojilerine hazırlamak için ümit vadetmektedir. Bu tezde, bilgi yaşını düşüren çizelgeleme problemini farklı ağ modellerinde inceledik. LCFS kuyruklarda Stochastic Hybrid System metoduyla yaptığımız teorik çalışmaların sonuçları yardımıyla Maximum Age Difference isimli ağları bilgi yaşı açısından optimize etmekte kullanılabilecek öne çıkan bir çizelgeleme algoritması önerdik. Ayrıca bilgi yaşı düşüren çizelgeleme problemini Reinforcement Learning tabanlı metodlarla birlikte inceledik. Bu yöntem özellikle ağın istatistiksel bilgilerini bilinmediği zaman avantajlı olmaktadır.

Anahtar Kelimeler: Bilgi Yaşı, Haberleşme Ağları, Çizelgeleme Algoritmaları

To my dear family...

"Man has to awaken to wonder - and so perhaps do people.
Science is a way sending him to sleep again."
Ludwig Wittgenstein

ACKNOWLEDGMENTS

First and foremost, I would first like to express my sincere appreciation and gratitude to my supervisor Prof. Dr. Elif Uysal for her continuous support, criticism and invaluable guidance throughout my thesis study. For their comments and criticism, I would also like to thank the examining committee members; Prof. Dr. Nail Akar, Prof. Dr. İlkay Ulusoy, Prof. Dr. Ali Özgür Yılmaz, Assist. Prof. Dr. Mustafa Mert Ankaralı.

I would like to express my gratitude to the friends from Communication Networks Research Group; Baran Tan Bacınoğlu, Sajjad Baghaee and Hakan Saç for easing my journey through my study by sharing their experience and providing guidance.

I would like to mention my friends, Furkan Karakaya, Halil İbrahim Uğurlu, Muhammed Yusuf Candan, Osman Kaan Karagöz, Fatih Uzgur and Fatih Çağatay Akyön for our friendly conversations on deep or small issues through my M.S. journey.

I am thankful to TÜBİTAK, the National Scientific and Technological Research Council of Turkey, for granting me their M.S. studies scholarship.

I am thankful to METU-BAP and TÜBİTAK for their financial support under 117E215 and 117E003.

I am thankful to BTK, the Information and Communication Technologies Authority, and Turk Telekom for granting me 5G and Beyond Program Fellowship.

Finally, but forever I owe my thanks to my family, my loving mother Melek, my dear father Mehmet and my brothers for their relentless support and encouragement, and especially I am deeply grateful to my beloved wife Duygu for her undying love and unfailing companion, without her support this work would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

AoI	Age of Information
IoT	Internet of Things
FCFS	First-Come-First-Served
LCFS	Last-Come-First-Served
MDP	Markov Decision Processes
SHS	Stochastic Hybrid System
TCP	Transport Control Protocol
IP	Internet Protocol
UDP	User Datagram Protocol
RTT	Round Trip Time
MAD	Maximum Age Difference
MAF	Maximum Age First
OP	Opportunistic Preemption
RL	Reinforcement Learning
DQN	Deep Q-Network
PG	Policy Gradients
CSMA	Carrier Sense Multiple Access
MGF	Moment Generating Function
ARQ	Automatic Repeat Request

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the emerging technologies, like IoT, smart cars, and various cloud services (e.g., cloud gaming), the number of devices connected to the Internet and the data traffic generated by various applications is increasing. Up until now, the expectation from service providers was Internet connection with high bandwidth and speed. High-speed Internet was sufficient to get video streams at high definition and downloading files quickly. However, today, the expectation of people and businesses is beyond the high-speed Internet connection. The number of sensors and IoT devices around us increasing day by day, and these devices are connected to transfer sensory data or control inputs for decision making, monitoring, and remote control.

Additionally, the data traffic generated by these devices is quite different from regular data we used to consume, like video streaming and web-surfing. Generally, this new type of data traffic composed of small data packets that carry *status updates*, which should be received by the destination as *freshly* as possible. For example, a smart car should receive information on the current situation of the road to avoid unnecessary traffic or an accident. Therefore, the information on the current situations carried by the status update packets should be delivered before the data gets stale and useless anymore. Meanwhile, we continue to generate and consume the regular type of data traffic, and these different types of data traffic will coexist in the future networks. An example of this coexistence is planned for 5G. In 5G, three different communication types are assumed: eMBB (enhanced mobile broadband), mMTC (massive machine-type communication) and URLLC (ultra-reliable low latency communication).

However, to prioritize different types of applications and measure the network performance, we need to measure the *freshness* provided by the communication link. An emerging metric *Age-of-Information* (AoI) is quite promising to quantify the freshness. Unlike previous metrics, the AoI takes into consideration not only the delay of the transmission but also the sampling process. Therefore, the AoI metric allows us to optimize not only the network but also the devices and applications connected to it.

The challenges mentioned above the future networks will face, require updates and improvements at all layers of the network architecture. In this thesis, we focus on scheduling mechanisms, which are essential to providing freshness and challenging due to the need to consider the diverse requirements of different applications. By exploiting the innovation brought on by the metric Age-of-Information, we propose new scheduling mechanisms that are quite useful for new generation networks.

1.2 Contributions

In this thesis, we examine scheduling scenarios on network models and the scheduling algorithms that improve Age-of-Information performance. The thesis presents analysis of age characteristics of LCFS-queues using the Stochastic Hybrid System approach, proposes an average-age optimal scheduling policy called Maximum Age Difference, and a Reinforcement Learning-based approach to minimize AoI in the real-life scheduling scenarios.

The contributions of the thesis can be listed as follows:

- A basic network model including two sources and LCFS queues with a buffer size of 1-packet is investigated under preemptive and nonpreemptive service disciplines using the SHS approach. We calculated the closed-form expressions for the average age of information. These results provide insight into LCFS characteristics in terms of AoI.

- An extension for the Stochastic Hybrid System technique is provided to analyze the network models with hypoexponential distributed service times. We also

provided the closed-form expressions for average age for the model with Erlang distributed service times. Some closed-form expressions are provided through the MATLAB script called Generic SHS Solver, developed by the author.

- The Maximum Age Difference policy, which is a scheduling algorithm, is proposed, and the optimality of this policy is proved for nonpreemptive service discipline.
- Opportunistic Preemption (OP) policy, which improves the preemption decision in LCFS queues, is proposed. The sufficient statistics of OP policy is computed for exponential and Erlang-2 distributed service times together with Poisson arrivals, simulation results provided for these setups.
- MAD-OP policy, which employs MAD policy for scheduling decisions and OP policy for preemption decisions, is proposed, and its performance under preemptive service discipline is shown with simulation results.
- A Reinforcement Learning-based approach is used to solve the age-minimizing scheduling problem, which is useful when the network model's statistics are unknown.

The material covered in the thesis have been published in [1–5].

1.3 The Outline of the Thesis

The rest of this thesis is organized as follows. In Chapter 2, the definition of Age of Information and some of the important studies in AoI literature are discussed. In Chapter 3, after providing a mathematical background on Stochastic Hybrid Systems, we used this method to analyze simple scheduling scenarios. In Chapter 4, the MAD policy is analyzed for preemptive and nonpreemptive service, and simulation results are provided. In Chapter 5, the scheduling problem is solved using a Reinforcement Learning approach. Finally, the conclusion of the thesis is given in Chapter 6.

CHAPTER 2

AGE OF INFORMATION

2.1 Definition of Age of Information

Although the age of a random process is a known concept [6], the metric *Age-of-Information* is first defined and used to analyze the timeliness of basic network models with FCFS and LCFS queues in [7–9].

Consider a simple model with a source node and a destination node. The source node receives new status update packet n at time s_n, and the interarrival time between the packets $(n-1)$ and n is denoted by X_n. The new arrivals wait in the queue of the source until the service, and S_n denotes the service time of packet n. Consider that status update packet arrival is modeled as a stochastic process with the arrival rate $\lambda = \frac{1}{\mathrm{E[X]}}$, and the packets are transmitted with average service rate $\mu = \frac{1}{\mathrm{E[S]}}$.

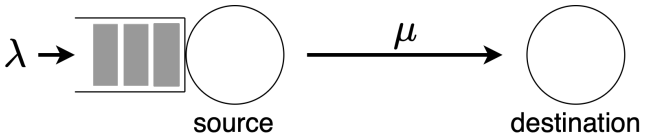

Figure 2.1: Simple system model

Given the system model Figure 2.1, the Age-of-Information is defined as below.

Definition 2.1.1. *Consider a communication link between a source and a destination. The status update arriving at the source at time s_n is received by the destination at time r_n. At time t, the index of the most recently received update is*

$$N(t) = \max\{n | r_n \leq t\}, \tag{2.1}$$

and the generation time of the most recent update is

$$U(t) = s_{N(t)}. \tag{2.2}$$

The Age of Information of the communication link between the source and the destination is a stochastic process, defined as

$$\Delta(t) = t - U(t). \tag{2.3}$$

Note that the terms *Age-of-Information*, *instantaneous age* or simply *age* are used interchangeably throughout the remaining of this thesis.

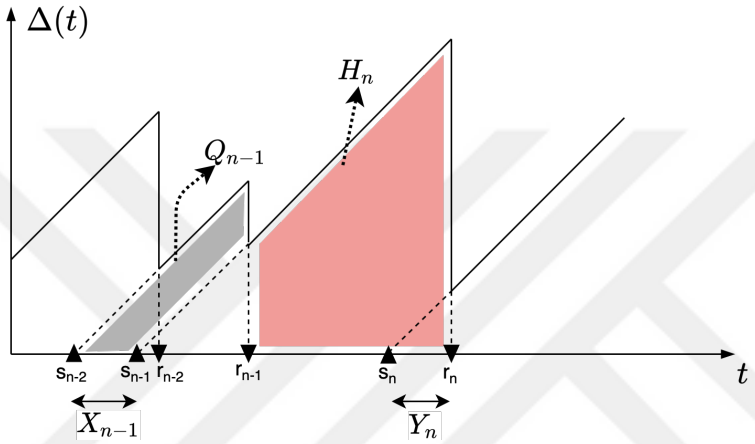

Figure 2.2: Sample path of the age process, s_n ans r_n denote the generation and delivery instant of packet n.

In Figure 2.2, a sample path of an age process is given. As seen in the figure, the age grows linearly with a slope of 1 unless there is no service completion. At the delivery instants r_n, the age process drops to Y_n, which is the time passed since the packet is generated. In other words, when the destination receives a new status update, the timeliness of the destination is as much as the timeliness of the received packet. During the period that the destination is not updated by the source, the *unawareness* of destination increases. Hence, the age process $\Delta(t)$ constitutes a sawtooth form.

Unlike the other metrics, like RTT or delay, the age is not computed per-packet but we study age process for the whole sequence of transmitted packets. Due to the formulation, the previous packet's delivery affects the age that the current packet encounters. Therefore, optimizing networks for AoI is more challenging and complex than other metrics.

In the literature, various forms of the AoI are studied, and policies developed based on those. The most common one is the time average age of information, which is computed as

$$\Delta_{\mathcal{T}} = \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} \Delta(t) dt, \tag{2.4}$$

and the steady-state time average age can be computed using the expected value of the area of the trapezoids Q_n or H_n, given in Figure 2.2, i.e.,

$$\lim_{\mathcal{T}\to\infty} \Delta_{\mathcal{T}} = \Delta = \frac{\mathrm{E[Q]}}{\mathrm{E[X]}} = \frac{\mathrm{E[H]}}{\mathrm{E[X]}}, \tag{2.5}$$

where

$$Q_n = \frac{1}{2}(2r_n - s_n - s_{n-1})(s_n - s_{n-1}), \tag{2.6a}$$

$$H_n = (r_n - r_{n-1})(r_{n-1} - s_{n-1}) + \frac{(r_n - r_{n-1})^2}{2}. \tag{2.6b}$$

Another metric that originated from AoI is the average peak Age of Information, which is defined as the mean of the ages just before the deliveries, in other words, the tips of the sawtooth age process, given as follows.

$$\Delta_{peak} = \lim_{\mathcal{T}\to\infty} \frac{1}{N(\mathcal{T})} \sum_{n=1}^{N(\mathcal{T})} \Delta(r_n^-). \tag{2.7}$$

In addition to these most common forms of AoI, in many works, such as [10–13], the age-penalty defined as the functional of age of information $f(\Delta(t))$, where $f : [0, \infty) \to \mathbb{R}$. Using different age-penalty functions enables us to define application-specific performance metrics related to age.

When one thinks about the Age of Information, the first question that comes into mind is the difference between delay and age. We can answer this question by using the early results on FCFS queues in [8]. In Figure 2.3, the average AoI and delay observed on an FCFS M/M/1 queue is illustrated. As seen in the figure, the average delay is low at low utilization since the queue is empty. However, since the packet arrivals are infrequent, a high average age is observed. On the other hand, at high utilization, since the queue is full and packets wait for a long time in queue, a high average age and delay are observed. Although the average delay is increasing with respect to utilization, the average age has a *bathtub shape*, which has an optimal sampling point

yielding the minimum average age. This is an interesting result because we need to operate at an under-utilized point ($\rho^* \approx 0.53$) to get the minimum average AoI. This "bathtub shaped" age graph is typical, especially for FCFS queues, and arouses the interest of researchers in the area.

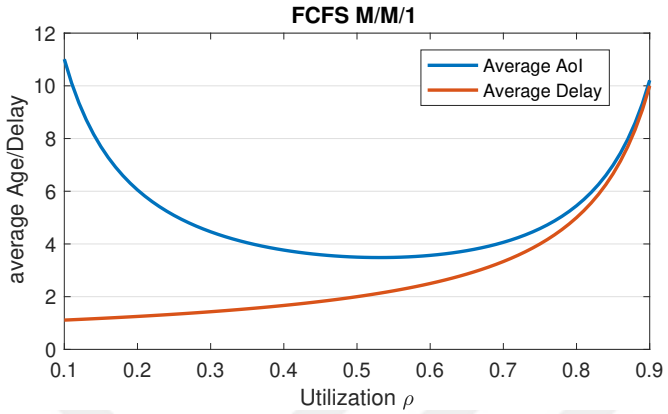

Figure 2.3: Average age and delay versus utilization on FCFS M/M/1 queue.

Now, we will provide the following definitions, which will be useful throughout this thesis.

Definition 2.1.2. ***Preemptive Service Discipline:*** *A service discipline is said to be preemptive if the ongoing transmission of a packet can be preempted upon the arrival of a new packet.*

Definition 2.1.3. ***Nonpreemptive Service Discipline:*** *A service discipline is said to be nonpreemptive if a new arrival should wait for the completion of the ongoing service to be served.*

Definition 2.1.4. ***Work-conserving Policy****: A policy is said to be work-conserving if the policy never imposes an idle duration when there is any job waiting.*

Definition 2.1.5. ***Age-effective transmission:*** *A transmission is said to be age-effective if the transmitted packet decreases the age process $\Delta(t)$ at the delivery. In other words, the timestamp of the transmitted packet should be higher than the timestamp of the most recently received packet.*

2.2 Related Works

The AoI performance of the communication link depends on many properties of the network. Therefore, many works in the literature focused on the evaluation of AoI or peak AoI in different system models to understand the factors affecting the AoI performance. As mentioned before, in the early works [8] the closed-form expressions for average AoI under FCFS M/M/1, D/M/1 and M/D/1 queue disciplines are evaluated. The advantages of LCFS discipline over FCFS discipline is investigated for preemptive and nonpreemptive service policies in [9]. Employing LCFS queues for minimizing age is a quite reasonable approach, as LCFS intrinsically prioritizes the fresh packets over old ones, and it helps to avoid the increase in average age at high utilization. Additionally, it is shown that preemption improves the age performance in many circumstances. However, as discussed in [14], the preemption is not age-optimal, especially for the models with Gamma distributed service times. As shown, the average age has a bathtub shape for the models with Poisson arrivals and Gamma service distribution, if the service is preemptive. Motivated by the observation that the packets get stale in the long queues, the effects of the buffer management techniques in FCFS queues, such as buffer size limitations and packet deadlines, are discussed in [15, 16]. A comprehensive overview can be found in [17], which also provides additional bounds and closed-form expressions for average age under various queuing disciplines.

Another interesting phenomenon revealed in [11, 18] points out the intuitive sampling policies, such as *zero-wait* policy, which generates the new status update just after the completion of the ongoing service, is away from the age-optimality under certain circumstances. The proposed policy in these studies waits before sampling for a short time to balance the early completion of the previous service, and the detailed conditions of the optimality of the proposed policy can be found in [11, 18].

The scheduling problem for networks with multiple sources is studied in [19–27]. The age-optimal wireless scheduling problem under physical interference is shown to be NP-hard in [20]. The scheduling problem for broadcast networks with multiple sources and a single base station is discussed in [22–24]. Among the various policies discussed in these works, a greedy policy that schedules based on the maximum

instantaneous age found to be optimal for many circumstances. This policy called Maximum Age First (MAF) in [25], and the optimality of this policy is proved under preemptive service discipline for synchronous sources, that receive the packet arrivals at the same time instants. In [26], the authors combine the results in [11,25], and consider the joint age-optimal scheduling and sampling problem. In studies [28–31], the age-minimization problem is examined in wireless networks under general interference constraints and channel uncertainty.

A useful technique called Stochastic Hybrid System (SHS) is introduced in [32], which enables us to compute the closed-form expression for relatively complex models with multiple sources and servers. In this study, SHS is used to analyze basic scheduling scenarios under LCFS service discipline with preemptive and nonpreemptive policies. Many other works have been published [32–43] using this technique for analyzing the networks with a single user or multiple users. In [36], by SHS technique, the moments and MGF of the age is calculated for various models. In [38, 44], the case of multiple sources, competing for a multiple access channel is studied under CSMA and an unslotted collision channel.

Another area related to our work is the conjecture of AoI and Reinforcement Learning. The first applications of reinforcement learning to the age-optimal scheduling problem appeared in [45, 46], which employed a variation of the SARSA algorithm for scheduling packet transmissions on a link using Hybrid ARQ. The model used in these studies adopted a *generate-at-will* model, which also used in [11], where the scheduler has control over the packet generation instants. The application of DQN to the age-optimal sampling problem has been considered in [47] on an experimental end-to-end IoT setup with a single source-destination pair running a TCP/IP connection. Another work [48] focused on the same optimal-sampling problem by using a model-based RL solution designed based on the theoretical result from [11]. Additionally, in this work, a model-free RL algorithm is proposed, which shows promising results under changing network conditions.

CHAPTER 3

CHARACTERISTICS OF LCFS POLICIES ON MULTI-USER NETWORKS

In this chapter, we analyzed average AoI on a 2-source network, that two sources with LCFS queues share the same channel. We proposed two policies, LCFS-Preemptive and LCFS-Nonpreemptive, that sources have their own LCFS queues with a buffer size of 1-packet, which are scheduled by a common server. We analyzed these models using the Stochastic Hybrid System (SHS) method, and obtained the closed-form expressions for average AoI. The results give us insight into how the LCFS queues perform in scheduling scenarios in terms of AoI. We also provided the achievable age regions for these scenarios.

Stochastic Hybrid System (SHS) method is a systematic approach to analyze a broad class of processes containing continuous and discrete dynamics. SHS is first utilized for age analysis in [49] and used to analyze models with multiple sources, different service policies and network scenarios [32–35, 37–43].

In Section 3.1, we provided a mathematical background for SHS and its utilization for average age analysis. In Section 3.2, we introduced LCFS-Preemptive and LCFS-Nonpreemptive models and their SHS models for exponential service distribution. We also obtained the closed-form expressions for these models. In Section 3.3, we proposed a framework that enables us to analyze the models with hypoexponential service distributions using the SHS method. We used this framework to obtain the closed-form expression not only for LCFS-Preemptive, LCFS-Nonpreemptive but also for LCFS-S and LCFS-W, proposed in [49], with Erlang service distribution, which is a special case of hypoexponential distribution.

3.1 Using Stochastic Hybrid System for Analyzing Average AoI

Stochastic Hybrid System is a method that enables us to analyze the continuous-time processes with Markovian characteristics. In the literature, there are various types of versions and formulations of SHS [50], but in this work, we embraced the model and notation in [49, 51]. The SHS technique proposed in [49] allows us to analyze AoI in various network models and scenarios. In subsection 3.1.1, we provide a brief introduction to SHS models, and in subsection 3.1.2 we explain the piecewise linear SHS model with linear reset functions for analyzing time average AoI.

3.1.1 Introduction to Stochastic Hybrid Systems

In an SHS model, a state is composed of a discrete component $q(t) \in \mathcal{Q} = \{0, \ldots, m\}$ and a continuous component $\mathbf{x}(t) = [x_0(t) \ \cdots \ x_n(t)] \in \mathbb{R}^{n+1}$. Given the k-vector $\mathbf{z}(t)$ of independent Brownian motion processes, an SHS is defined by a stochastic differential equation

$$\dot{\mathbf{x}} = f(q, \mathbf{x}, t) + g(q, \mathbf{x}, t)\dot{\mathbf{z}} \tag{3.1}$$

where $f : \mathcal{Q} \times \mathbb{R}^{n+1} \times [0, \infty) \to \mathbb{R}^{n+1}$ and $g : \mathcal{Q} \times \mathbb{R}^{n+1} \times [0, \infty) \to \mathbb{R}^{(n+1)\times k}$,

a set of transitions $\mathcal{L} = \{1, \ldots, \ell\}$, such that each $l \in \mathcal{L}$ defines a discrete transition/reset map

$$(q', \mathbf{x}') = \phi_l(q, \mathbf{x}, t), \quad \phi_l : \mathcal{Q} \times \mathbb{R}^{n+1} \times [0, \infty) \to \mathcal{Q} \times \mathbb{R}^{n+1} \tag{3.2}$$

and transition intensity

$$\lambda^{(l)}(q, \mathbf{x}, t), \quad \lambda^{(l)} : \mathcal{Q} \times \mathbb{R}^{n+1} \times [0, \infty) \to [0, \infty) \tag{3.3}$$

While the system is in discrete state q, the continuous state $\mathbf{x}(t)$ grows according to (3.1). When a transition $l \in \mathcal{L}$ occurs from discrete state q to q', depending on the transition/reset map given in (3.2), the continuous state can have a discontinuous jump which is subject to the current discrete and continuous state and time. Therefore, the continuous state $\mathbf{x}(t)$ is a piecewise continuous process. For each transition

$l \in \mathcal{L}$ there is a counting process $N_l(t)$ that counts the number of times that the corresponding transition/reset map ϕ_l is activated in the interval $[0, t]$. The probability of increment in N_l in the interval $(t, t + dt]$ is $\lambda^{(l)}(q, \mathbf{x}, t)dt$.

As suggested in [51], the expected values of test functions $\psi(q, \mathbf{x}, t)$ can be used to obtain statistical properties of continuous state $\mathbf{x}(t)$. These test functions are used in [49] to generate the closed-form expression of average AoI for various network scenarios. In subsection 3.1.2, analyzing model for average AoI by using the piecewise linear SHS model with linear reset functions is explained. For additional background information on SHS, [50, 51] can be examined.

3.1.2 Stochastic Hybrid System to Analyze AoI

To model the sawtooth age process using SHS system, we need a subclass of SHS models that can cover the linear increases in discrete states and discontinuous jumps in transitions. Therefore, we focus on the case where $q(t)$ is a continuous-time finite-state Markov chain with memoryless transitions between the discrete states and $\mathbf{x}(t) \in \mathbb{R}^{n+1}$ describes the continuous time evolution of the collection of age-related processes. To capture the age process we restrict the general SHS model defined by (3.1) and (3.2) as given below.

$$f(q, \mathbf{x}, t) = \mathbf{b}_q \tag{3.4a}$$

$$g(q, \mathbf{x}, t) = 0 \tag{3.4b}$$

$$\lambda^{(l)}(q, \mathbf{x}, t) = \lambda^{(l)} \delta_{q_l, q} \tag{3.4c}$$

$$\phi_l(q, \mathbf{x}, t) = (q'_l, \mathbf{x}\mathbf{A}_l) \tag{3.4d}$$

The graphical representation of SHS model we use throughout this chapter is different from [51]. In the graphical representation we use, each state of Markov chain $q \in \mathcal{Q}$ is depicted as a node and each transition l is depicted as a directed edge (q_l, q'_l) with transition $\lambda^{(l)}$ while $q(t) = q_l$. In other words, q_l indicates currently active state and q'_l is the following state after the transition l. Kronecker delta function $\delta_{q_l, q}$ in (3.4c) allows the transition l to occur only in state q_l. Considering (3.4a) and (3.4b), the continuous state evolution (3.1) in each state $q(t) = q$ becomes

$$\dot{\mathbf{x}} = \mathbf{b}_q \tag{3.5}$$

where $\mathbf{b}_q \in \mathbb{R}^{n+1}$ with the same size of $\mathbf{x}(t)$, stating the constant slope of each component of $\mathbf{x}(t)$ for each state $q \in \mathcal{Q}$.

Furthermore, the transition l occurred according to a transition/reset map (3.4d) causes a discontinuous jump in continuous state $\mathbf{x}$ which is a linear mapping of the form $\mathbf{x}' = \mathbf{x}\mathbf{A}_l$ and a transition from discrete state q_l to q'_l.

Additionally, the transition link l can be describe by a 4-tuple $a_l = \left(q_l, q'_l, \lambda^{(l)}, \mathbf{A}_l\right)$ and the set of transitions is $\mathcal{A} = \{a_l : l \in \mathcal{L}\}$. Furthermore, the changes in continuous state $\mathbf{x}(t)$ in each state can be defined as $\mathcal{B} = \{\mathbf{b}_q : q \in \mathcal{Q}\}$. Therefore, a piecewise linear SHS with linear reset maps can be specified by the 3-tuple $(\mathcal{Q}, \mathcal{B}, \mathcal{A})$.

SHS associates the continuous state $\mathbf{x}(t)$ with the discrete states of a continuous time Markov chain $q \in \mathcal{Q}$. The transitions cause change not only in the discrete state but also in the continuous state. Therefore, a transition link l has an exponential transition rate $\{\lambda^{(l)}\}$ together with a linear transformation matrix $\{\mathbf{A}_l\}$. This association brings same differences between an SHS model and an ordinary continuous time Markov chain. Although the self-transitions are meaningless in an ordinary continuous time Markov chain, they cannot be omitted, if the self-transitions change the continuous state $\mathbf{x}(t)$ of the SHS model. Additionally, in an ordinary continuous-time Markov chain, between the discrete states $i, j \in \mathcal{Q}$ there is only one transition, but in case of an SHS, two transitions l and l' from i to j should be distinguished if linear reset matrices $\mathbf{A}_l$ and $\mathbf{A}_{l'}$ are different.

As mentioned above, to analyze stochastic characteristics of the continuous process $\mathbf{x}(t)$, [51] uses test functions. For age analysis, we adopt the test functions defined in [49]. For each $\bar{q} \in \mathcal{Q}$, there are two test functions defined as

$$\psi_{\bar{q}}(q, \mathbf{x}) = \delta_{\bar{q},q} \tag{3.6a}$$

and

$$\psi_{\bar{q}j}(q, \mathbf{x}) = x_j \delta_{\bar{q},q}, \quad j \in [0, n] \tag{3.6b}$$

The expectations of these test functions are defined for all $\bar{q} \in \mathcal{Q}$,

$$\pi_{\bar{q}}(t) = \mathrm{E}\left[\psi_{\bar{q}}(q(t), \mathbf{x}(t))\right] = \mathrm{E}\left[\delta_{\bar{q},q(t)}\right] \tag{3.7a}$$

$$v_{\bar{q}j}(t) = \mathrm{E}\left[\psi_{\bar{q}j}(q(t), \mathbf{x}(t))\right] = \mathrm{E}\left[x_j(t)\delta_{\bar{q},q(t)}\right], \quad j \in [0, n] \tag{3.7b}$$

and the vector function is defined accordingly,

$$\mathbf{v}_{\bar{q}}(t) = [v_{\bar{q}0}(t), \ldots, v_{\bar{q}n}(t)] = \mathrm{E}\left[\mathbf{x}(t)\delta_{\bar{q},q(t)}\right]$$

Note that $\pi_{\bar{q}}(t)$ in (3.7a) denotes the discrete Markov state probabilities

$$\pi_{\bar{q}}(t) = \mathrm{E}[\delta_{\bar{q},q(t)}] = \mathrm{P}[q(t) = \bar{q}]. \tag{3.8}$$

According to [51], a mapping $\psi \rightarrow L\psi$ is called as the extended generator of the SHS, and the extended generator of a piecewise linear SHS is given by

$$(L\psi)(q, \mathbf{x}) = \frac{\partial \psi(q, \mathbf{x})}{\partial \mathbf{x}} \cdot \mathbf{b}_q + \sum_{l \in \mathcal{L}} \left(\psi\left(\phi_l(q, x)\right) - \psi(q, x)\right) \lambda^{(l)}(q). \tag{3.9}$$

We note that each test function $\psi(q(t), \mathbf{x}(t))$ must satisfy Dynkin's formula

$$\frac{d\mathrm{E}[\psi(q(t), \mathbf{x}(t))]}{dt} = \mathrm{E}[(L\psi)(q(t), \mathbf{x}(t))]. \tag{3.10}$$

In [49, Lemma 2], it is shown that the test functions $\pi_{\bar{q}(t)}$ and $\mathbf{v}_{\bar{q}(t)}$ obey the first order differential equations given as

$$\dot{\pi}_{\bar{q}}(t) = \sum_{l \in \mathcal{L}'} \lambda^{(l)} \pi_{q_l}(t) - \pi_{\bar{q}}(t) \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)}, \quad \bar{q} \in \mathcal{Q}, \tag{3.11a}$$

$$\dot{\mathbf{v}}_{\bar{q}}(t) = \mathbf{b}_{\bar{q}} \pi_{\bar{q}}(t) + \sum_{l \in \mathcal{L}'_{\bar{q}}} \lambda^{(l)} \mathbf{v}_{q_l}(t) \mathbf{A}_l - \mathbf{v}_{\bar{q}}(t) \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)}, \quad \bar{q} \in \mathcal{Q}, \tag{3.11b}$$

where $\mathcal{L}'_{\bar{q}} = \{l \in \mathcal{L} : q'_l = \bar{q}\}$ and $\mathcal{L}_{\bar{q}} = \{l \in \mathcal{L} : q_l = \bar{q}\}$ denote the sets of incoming and outgoing transition for each state $\bar{q}$, respectively.

Since $\mathbf{v}_{\bar{q}}(t) = \mathrm{E}[\mathbf{x}(t)\delta_{\bar{q},q(t)}]$, the expected value of the continuous state is

$$\mathrm{E}[\mathbf{x}(t)] = \sum_{\bar{q} \in \mathcal{Q}} \mathrm{E}\left[\mathbf{x}(t)\delta_{\bar{q},q(t)}\right] = \sum_{\bar{q} \in \mathcal{Q}} \mathbf{v}_{\bar{q}}(t), \tag{3.12}$$

and (3.11) enable us to compute (3.12).

In the section 3.2, we examine LCFS-Preemptive and LCFS-Nonpreemptive scheduling algorithms using piecewise SHS model and framework proposed in [49], in which the continuous state $\mathbf{x}(t) = [x_0(t) \ \dots \ x_n(t)]$ track the age-related values. The first element $x_0(t)$ is the age process of the source of interest, and the other elements $x_i(t)$ for $i \in \{1, \ \dots \ , n\}$ keep the age value of source of interest at the sampling instant $x_0(t_s)$ of i^{th} packet waiting service completion. Note that in an LCFS queue, $x_1(t)$ actually stores the amount of decrease in the age process at the delivery of it. The evolution of the continuous state's elements at sampling and the delivery of a packet can be shown as

$$x_1(t_s) = x_0(t_s), \tag{3.13a}$$

$$x_0(t_d) = x_0(t_d^-) - x_1(t_s) \quad \text{for } t_d > t_s, \tag{3.13b}$$

where t_s and t_d are the sampling and the delivery times, respectively.

For the given continuous state $\mathbf{x}(t)$, we see that $x_i(t), \ 1 \leq i \leq n$ is constant while the system is in a discrete state $q \in \mathcal{Q}$. However, as $x_0(t)$ tracks the age process, it increases with a constant slope of 1, while the system is in $q \in \mathcal{Q}$. Therefore, the slope vector $\mathbf{b}_q$ in (3.5) is set

$$[\mathbf{b}_q]_j = \begin{cases} 1, & \text{for } j = 0 \\ 0, & \text{for } j \in \{1, \ \dots \ , n\} \end{cases}, \forall q \in \mathcal{Q}, \tag{3.14}$$

where $[\cdot]_j$ denotes the jth column of corresponding matrix.

In this model, the transitions corresponding the events such as service completion of a packet or the arrival of a new packet, can be described with a linear combination of the elements of continuous state $\mathbf{x}(t)$ (3.13), and corresponding transition/reset maps are $\mathbf{A}_l \in \{-1, 0, 1\}^{(n+1)(n+1)}$. The set of linear mappings $\{\mathbf{A}_l\}$ depends on the specific queue discipline and network scenario.

Definition 3.1.1. *An age-of-information SHS $(\mathcal{Q}, \mathcal{B}, \mathcal{A})$ is an SHS in which discrete state $q(t) \in \mathcal{Q}$ is a continuous-time Markov chain with transitions $l \in \mathcal{L}$ from state q_l to q_l' at rate $\lambda^{(l)}$, and the continuous state evolves according to $\dot{\mathrm{x}}(t) = \mathrm{b}_q \in \{0, 1\}^{n+1}$ in each discrete state $q \in \mathcal{Q}$ and is subject to the linear transition reset map $\mathrm{x}' = \mathrm{x}\mathrm{A}_l$ in transition l.*

With this SHS model, time-average age analysis is possible if the continuous-time Markov chain $q \in \mathcal{Q}$ is ergodic. Under this assumption, we can say that a unique stationary state probability vector $\bar{\boldsymbol{\pi}} = [\bar{\pi}_0 \ \dots \ \bar{\pi}_m]$ exists and it satisfies

$$\begin{aligned} \bar{\pi}_{\bar{q}} \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)} &= \sum_{l \in \mathcal{L}'_{\bar{q}}} \lambda^{(l)} \bar{\pi}_{q_l} \\ \sum_{\bar{q} \in \mathcal{Q}} \bar{\pi}_{\bar{q}} &= 1 \end{aligned} \tag{3.15}$$

When stationary state probability vector $\bar{\boldsymbol{\pi}}$ exists, the differential equation (3.11b) reduces to

$$\dot{\mathbf{v}}_{\bar{q}}(t) = \mathbf{b}_{\bar{q}} \bar{\pi}_{\bar{q}} + \sum_{l \in \mathcal{L}'_{\bar{q}}} \lambda^{(l)} \mathbf{v}_{q_l}(t) \mathbf{A}_l - \mathbf{v}_{\bar{q}}(t) \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)}, \quad \bar{q} \in \mathcal{Q} \tag{3.16}$$

Although the Markov chain $q \in \mathcal{Q}$ is ergodic, it does not ensure that the (3.16) is stable. The stability of (3.16) depends on the set of transition/reset maps $\{\mathbf{A}_l\}$. When (3.16) is stable, as $t \to \infty$ each $\mathbf{v}_{\bar{q}}(t) = \mathrm{E}[\mathbf{x}(t)\delta_{\bar{q},q(t)}]$ approaches a limit $\bar{\mathbf{v}}_{\bar{q}}$. In this case, it follows from (3.12) that

$$\mathrm{E}[\mathbf{x}] = \lim_{t \to \infty} \mathrm{E}[\mathbf{x}(t)] = \lim_{t \to \infty} \sum_{\bar{q} \in \mathcal{Q}} \mathrm{E}\left[\mathbf{x}(t)\delta_{\bar{q},q(t)}\right] = \sum_{\bar{q} \in \mathcal{Q}} \bar{\mathbf{v}}_{\bar{q}} \tag{3.17}$$

Note that the average age of the process of interest is $\Delta = \lim_{t \to \infty} \mathrm{E}[x_0(t)] = \sum_{\bar{q} \in Q} \bar{v}_{\bar{q}0}$. These limiting values can be calculated by setting the derivatives $\dot{\pi}_{\bar{q}}(t)$ and $\dot{\mathbf{v}}_{\bar{q}}(t)$ in (3.11) to zero and solving for the limiting values $\bar{\pi}_{\bar{q}}$ and $\bar{\mathbf{v}}_{\bar{q}}$.

Theorem 1 ([49]). *If the discrete-state Markov chain $q(t)$ is ergodic with stationary state probability $\bar{\boldsymbol{\pi}}$ and we can find a non-negative solution $\bar{\mathbf{v}} = [\bar{\mathbf{v}}_0 \ \dots \ \bar{\mathbf{v}}_m]$ such that*

$$\bar{\mathbf{v}}_{\bar{q}} \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)} = \mathbf{b}_{\bar{q}} \bar{\pi}_{\bar{q}} + \sum_{l \in \mathcal{L}'_{\bar{q}}} \lambda^{(l)} \bar{\mathbf{v}}_{q_l} \mathbf{A}_l, \quad \bar{q} \in \mathcal{Q} \tag{3.18}$$

then the differential equation (3.16) *is stable and the average age of an AoI SHS model is given by*

$$\Delta = \sum_{\bar{q} \in \mathcal{Q}} \bar{v}_{\bar{q}0}. \tag{3.19}$$

Although the equations are described explicitly, even small extensions to minimal queuing models may drastically increase the number of transitions and discrete states, and the complexity of the solution. For simple SHS models, it is quite easy to solve (3.16), but in many models, we will discuss in the next sections require computational tools. We developed a framework on MATLAB, called Generic SHS Solver [1] to solve the AoI SHS models defined according to the Definition 3.1.1.

In section 3.2, we discussed an AoI SHS model that is an extended version of the models discussed in [49], with the difference of sources having separate 1-packet LCFS buffers. In section 3.3, we propose a framework that enables us to analyze network models with hypoexponential service distribution using SHS technique. The numerical results are provided for Erlang service distribution, which is a special case of hypoexponential distribution.

3.2 LCFS Scheduling under Exponentially Distributed Service Times

The LCFS-S and LCFS-W models in [49] characterize the case that multiple sources receive new packets based on independent Poisson processes and new packets wait in an LCFS buffer of a joint server with a buffer size of 1-packet; the packet in the buffer is served according to preemptive and nonpreemptive policies. These scenarios are eligible to analyze using SHS approach, with relatively small models, having one discrete state for LCFS-S and three discrete states for LCFS-W.

In our study, we slightly changed these models. In our models, each source has a 1-packet LCFS buffer, and a common server serves the sources according to LCFS-Preemptive and LCFS-Nonpreemptive policies. With these models, we analyze the effect of having individual buffers on the average age-of-information. Note that the models mentioned above can be considered as basic scheduling scenarios. Therefore, in the rest of the chapter, we will consider that the policies are scheduling algorithms, and they are defined as follows.

[1] Generic SHS Solver is a MATLAB tool, developed by the author. It provides a framework to define the SHS models, and it generates the close-form expression of average age for the given network model. It is available at github.com/hbbeytur/GenericSHSsolver

Definition 3.2.1. *LCFS-Preemptive and LCFS-Nonpreemptive are scheduling algorithms decide the next source $k^* \in \{1, 2\}$ to be served according to the rule*

$$k^* = \arg\max_k \{S_k(t) : S_k(t) \leq t\}, \quad k \in \{1, \ \dots \ , K\}, \tag{3.20}$$

where $S_k(t)$ is the timestamp of the age-effective packet waiting in the k^{th} source's buffer, and the service policy is preemptive and nonpreemptive, respectively.

Note that the SHS models we built for these policies consider that each source has own buffer of size 1-packet. Therefore, unlike LCFS-S and LCFS-W, that assume a common buffer, our SHS models with 2-sources cannot be generalized for multiple sources. To analyze the policies for more than two sources, other SHS models with more discrete states has to be studied.

In the rest of the section, we will investigate these algorithms using SHS approach, on a network model having two sources with separate 1-packet buffers.

3.2.1 LCFS-Preemptive Scheduling

The SHS model representing the LCFS-Preemptive algorithm has five discrete states $q \in \mathcal{Q} = \{0, 1, 2, 3, 4\}$ such that $q = 0$ denotes that the server is idle, $q \in \{1, 2\}$ indicates the source of the update packet in service, while other source does not have any packet, and in the state $q \in \{3, 4\}$, while the source 1 (source 2) has a packet in service, source 2 (source 1) has a preempted packet in its buffer, which is still age-effective.

The continuous state is $\mathbf{x}(t) = [x_0(t) \ x_1(t) \ x_2(t)]$, where $x_0(t)$ is the age process of source 1, $x_1(t)$ is the amount of the reduction in age process of source 1 to occur when the ongoing service is completed, and $x_2(t)$ is the amount of the reduction in age process of source 1 to occur when the next service is completed.

In Figure 3.1, a graphical representation of SHS model for LCFS-Preemptive Scheduling with two sources can be seen. Unlike a regular continuous-time Markov chain representation, the transitions are given with a tuple of $(\lambda^{(l)}, \mathbf{x}\mathbf{A}_l)$. Although this representation enough to understand the dynamics of an SHS model, we provide a

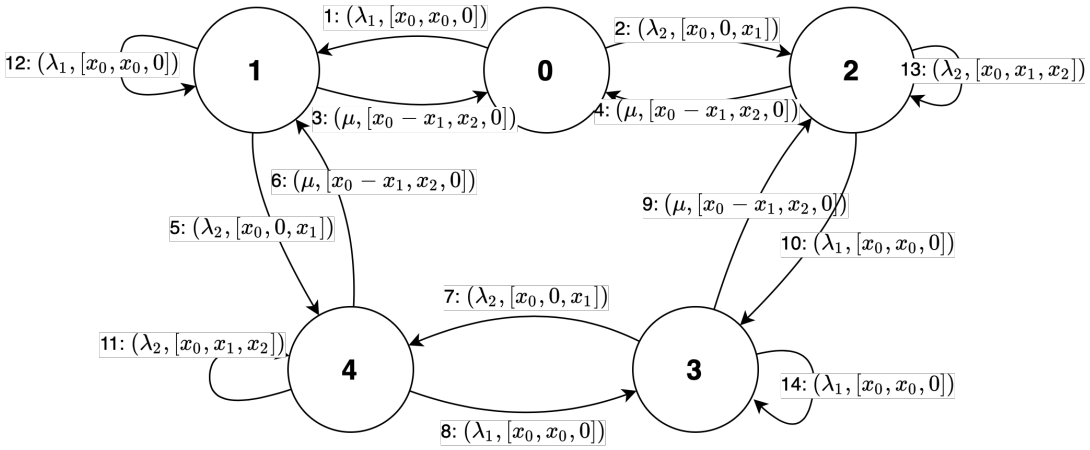

Figure 3.1: SHS model for LCFS-Preemptive Scheduling, the nodes and directed edges illustrate the discrete states and transitions between nodes. The detailed information on transitions is given in Table A.1.

detailed table containing the tuple $a_l = (q_l, q_l', \lambda^{(l)}, \mathbf{A}_l)$ for each transition l in Table A.1. Additionally, we provided $\mathbf{x}' = \mathbf{x}\mathbf{A}_l$ and $\mathbf{v}_{q_l}\mathbf{A}_l$ for convenience.

Before explaining each transition l, we note that to be able to reduce the SHS model in the way discussed in subsection 3.2.2, we keep the number of distinct transitions defined by the combination of transition rate $\lambda^{(l)}$ and transition/reset map $\mathbf{A}_l$ at minimum. With this approach, we manage to model the system with a minimum number of *events* occur during transitions, which are explained in the following. As seen in Figure 3.1, each transition in the model can be identified by one of 4 distinct events.

Now let us explain these four events:

1st Event, "$(\lambda_1, [x_0, x_0, 0])$": This event indicates the arrival of a new packet to source 1, and it occurs with an exponential rate of λ_1. Since the service policy is preemptive, a new arrival preempts the ongoing service. Therefore, the continuous state $x_1(t)$ is reset to the instantaneous age of source 1 $x_0(t)$. $x_2(t)$ is reset to 0 since the preempted packet is no longer age-effective, and its age reduction is not needed to be stored. Additionally, as the new arrival does not change the instantaneous age, $x_0(t)$ does not change and continues to increase with a constant slope of $[\mathbf{b}_q]_0 = 1$. This event occurs in every discrete state q. In discrete state $q = 0$, the idle server receives a new packet originated from source

1; consequently, a jump occurs to state $q' = 1$. In state $q = 1$, the new arrival preempts the ongoing service and restarts the service for source 1. Hence, the discrete state does not change. In state $q = 2$, a new arrival preempts the ongoing service for source 2 and transition occurs to $q' = 3$. In state $q = 3$, a new arrival to source 1 restarts the service of source 1, and the discrete state stays the same. Note that in this state, there is a packet waiting in source 2's buffer. Similar to state $q = 2$, in state $q = 4$, this event causes the preemption of the service of source 2, and a transition occurs to state $q' = 3$.

2nd Event, "$(\lambda_2, [x_0, 0, x_1])$": This event denotes the arrival of a new packet to source 2, and it occurs with an exponential rate λ_2. Since the service policy in this model is preemptive, any new arrival causes a preemption of the ongoing service. Similarly, a new arrival to source 2 does not change the age process of source 1, which is denoted by $x_0(t)$. The continuous state $x_1(t)$ is reset to zero since the service started with the new arrival to source 2 is not age-effective for source 1. However, the continuous state $x_2(t)$ changes differently; it is reset to $x_1(t)$ to capture the age value related to the preempted packet of source 1 if it exists. This event happens in discrete state $q = \{0, 1, 3\}$. In state $q = 0$, the idle server encounters a new packet of source 2 and starts to serve it. Consequently, a transition takes place to state $q' = 2$. In state $q = 1$, the event causes a preemption of source 1's packet, and the discrete state becomes $q' = 4$. Likewise, in state $q = 3$, source 1's packet is preempted, and a transition occurs to $q' = 4$.

3rd Event, "$(\lambda_2, [x_0, x_1, x_2])$": This events also denotes a new arrival to source 2, but it occurs only in the discrete states, where a service for source 2 goes on, which are $q = \{2, 4\}$. Note that this event causes a self-transition. In an SHS model, the self-transition is meaningful if the self-transition changes the continuous state. But in this event, since the new arrival to source 2 is not age-effective for source 1, it does not change the continuous state $\mathbf{x}(t)$. Therefore, these self-transitions in discrete state $q = \{2, 4\}$ can be omitted. For completeness, we put these transitions on the graphical model, as well.

4th Event, "$(\mu, [x_0 - x_1, x_2, 0])$": This event denotes the service completion for either source 1 or source 2, and it occurs with an exponential rate μ. When a service for source 1 is completed, the age process drops. Therefore, upon this event, a

reset occurs in continuous state x_0 such $x_0' = x_0 - x_1$. Note that x_1 is not 0, if and only if the completed service is for source 1. Additionally, since the service policy is work-conserving, if there is any packet waiting in the buffer, the server serves that packet. Therefore, continuous state x_1 is reset to x_2, which stores the amount of reduction in source 1's age process upon the service completion of the waiting packet. Again, x_2 is non-zero if and only if the completed service belongs to source 2 and there is a packet waiting in source 1's buffer. This event occurs in all discrete states except $q = 0$. In state $q = 1$, upon the occurrence of this event, the age of source 1 is reduced by the amount of x_1, which is non-zero, and the next discrete state is $q' = 0$. However, since there is not any waiting packet, x_2 is zero. Therefore, after the event x_1 is reset to zero. In state $q = 2$, the ongoing service belongs to source 2 and there is not any waiting packet. Hence, x_1 and x_2 are both zero. Because of this, the transition changes only the discrete state to $q' = 0$. In state $q = 3$, since the ongoing service belongs to source 1, upon the transition, x_0 is reduced by the non-zero amount of x_1 and the system jumps to state $q' = 2$. Additionally, since the waiting packet belongs to source 2 and it is not age-effective for source 1, x_1 is reset to 0 by $x_1' = x_2$. In discrete state $q = 4$, the ongoing service belongs to source 2 and there is a waiting packet in source 1's buffer. Therefore, age process of source 1 x_0 stays constant, and $x_1' = x_2$, where x_2 is non-zero.

The self-transitions are meaningful in SHS models but for some cases self-transitions do not affect the SHS model's solution as in $l = \{11, 13\}$ in Figure 3.1. We will clarify this phenomenon with the following proposition.

Proposition 1. *In an AoI SHS, in a discrete state $q \in \mathcal{Q}$, a self-transition $l \in \mathcal{L}$ with any exponential rate $\lambda^{(l)}$ and with the identity reset map $\mathbf{A}_l = \mathbf{I}$ can be discarded.*

As discussed above, many discrete states in Figure 3.1 have similar transitions. Using these similarities between the states, it is actually not hard to see that a smaller SHS model can represent the same system. Instead of solving the SHS model in Figure 3.1, we want to go on with the reduced SHS models of the same network scenario. In the next section, we will discuss how an SHS model can be reduced.

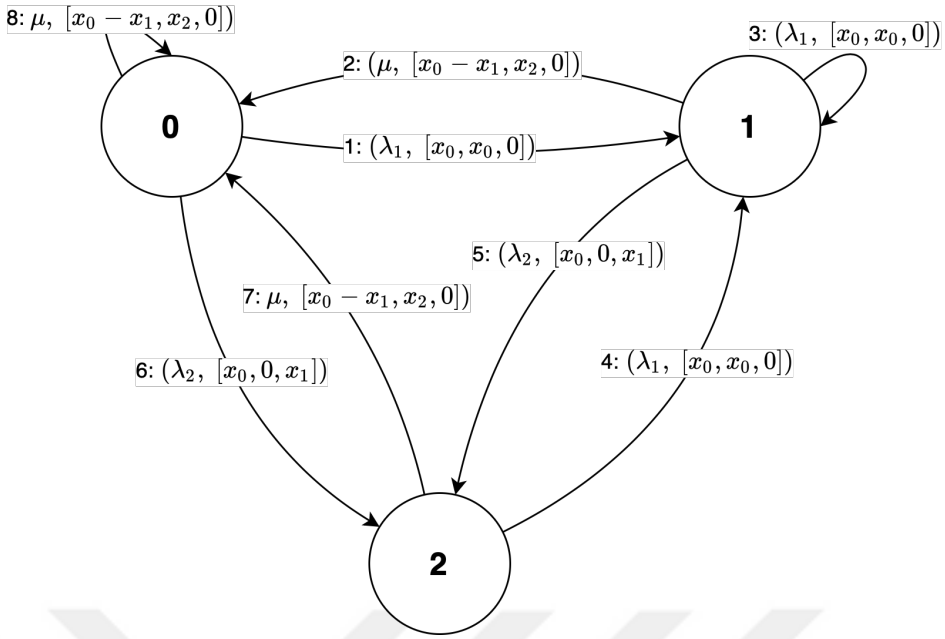

Figure 3.2: A simplified SHS model for LCFS-Preemptive Scheduling. The detailed information on transitions is given in Table A.2.

3.2.2 Reducing Stochastic Hybrid Systems

In general, the discrete states of an SHS model correspond to specific cases within the network model. For example, in the LCFS-Preemptive Scheduling model, the discrete states $q = \{0, 1, 2, 3, 4\}$ are defined for different conditions of the queues and services. Therefore, many SHS models can be illustrated by a smaller model by redefining the discrete states.

In Figure 3.2, LCFS-Preemptive Scheduling algorithm is represented by using a simplified SHS model with a smaller discrete state set $\mathcal{Q} = \{0, 1, 2\}$. The discrete states are described as follows

- $(q = 0)$: The server is idle, or serving previously preempted packet.
- $(q = 1)$: The server is serving a fresh packet of source 1.
- $(q = 2)$: The server is serving a fresh packet of source 2.

Note that when the system is in state $q = 0$, the server may be serving a preempted

packet that belongs to either source 1 or source 2. The event set constitutes the transitions in the simplified model (Figure 3.2) is the same as the event set of the original SHS model (Figure 3.1). Therefore, we refer the reader to the events listed in 3.2.1 for detailed information on transitions.

Although in this example, we illustrate the same scenario with a smaller discrete state set $\mathcal{Q}$, in many cases reducing the SHS model is not straightforward. In the following, we propose an analytical way of reducing SHS model, which does not depend on stating discrete states based on some heuristics.

Theorem 2. *In an AoI SHS with $(\mathcal{Q}, \mathcal{B}, \mathcal{A})$, if each discrete state $q \in \mathcal{Q}$ satisfies*

$$\alpha = \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)}, \quad \alpha \in \mathbb{R}, \tag{3.21}$$

and all discrete states $q \in \hat{\mathcal{Q}} \subseteq \mathcal{Q}$ have the same transition rate $\lambda^{(l)}$ and $\mathbf{A}_l$ for each $l \in \mathcal{L}_q$, the SHS model with $(\mathcal{Q}', \mathcal{B}', \mathcal{A}')$ yields the same average age-of-information

$$\Delta = \sum_{\bar{q} \in \mathcal{Q}'} \bar{v}_{\bar{q}0} = \sum_{\bar{q} \in \mathcal{Q}} \bar{v}_{\bar{q}0}, \tag{3.22}$$

where $\mathcal{Q}' = (\mathcal{Q} \setminus \hat{\mathcal{Q}}) \cup \hat{q}$, and $\hat{q}$ is the compound discrete state of $q \in \hat{\mathcal{Q}}$.

Proof. With the definition of $\mathcal{L}_{ij} = \{l \in \mathcal{L} : q_l = i, q'_l = j\}, \quad i, j \in \mathcal{Q}$, we can rewrite (3.18) as

$$\bar{\mathbf{v}}_j \sum_{l \in \mathcal{L}_j} \lambda^{(l)} = \mathbf{b}_j \bar{\pi}_j + \sum_{i \in \mathcal{Q}} \bar{\mathbf{v}}_i \sum_{l \in \mathcal{L}_{ij}} \lambda^{(l)} \mathbf{A}_l, \qquad i, j \in \mathcal{Q} \tag{3.23}$$

When we sum (3.23) over $j \in \mathcal{Q}$, we get

$$\sum_{j \in \mathcal{Q}} \bar{\mathbf{v}}_j \sum_{l \in \mathcal{L}_j} \lambda^{(l)} = \sum_{j \in \mathcal{Q}} \mathbf{b}_j \bar{\pi}_j + \sum_{j \in \mathcal{Q}} \sum_{i \in \mathcal{Q}} \bar{\mathbf{v}}_i \sum_{l \in \mathcal{L}_{ij}} \lambda^{(l)} \mathbf{A}_l, \qquad i, j \in \mathcal{Q} \tag{3.24}$$

Given that $\alpha = \sum_{l \in L_{\bar{q}}} \lambda^{(l)}, \ \forall \bar{q} \in \mathcal{Q}$, and with the observation that summation over $i \in \mathcal{Q}$ and $j \in \mathcal{Q}$ is same, (3.24) becomes

$$\alpha \sum_{i \in \mathcal{Q}} \bar{\mathbf{v}}_i = \sum_{i \in \mathcal{Q}} \mathbf{b}_i \bar{\pi}_i + \sum_{i \in \mathcal{Q}} \bar{\mathbf{v}}_i \sum_{j \in \mathcal{Q}} \sum_{l \in \mathcal{L}_{ij}} \lambda^{(l)} \mathbf{A}_l, \qquad i, j \in \mathcal{Q} \tag{3.25}$$

If the transition rates $\lambda^{(l)}$ and reset maps $\mathbf{A}_l$ of transition $l \in L_q$ for each $q \in \hat{\mathcal{Q}} \subseteq \mathcal{Q}$ are same, we can substitute the discrete states $q \in \hat{\mathcal{Q}}$ with a new discrete state $\hat{q}$,

which yields $\sum_{i\in\hat{\mathcal{Q}}} \bar{\mathbf{v}}_i = \bar{\mathbf{v}}_{\hat{q}}$, and every outgoing transition $l \in L_{\hat{q}}$ has target state $\hat{q}'_l = q'_l$ if $q'_l \in \hat{\mathcal{Q}}$. Consequently,

$$\alpha(\bar{\mathbf{v}}_{\hat{q}} + \sum_{i\in\mathcal{Q}\setminus\hat{\mathcal{Q}}} \bar{\mathbf{v}}_i) =$$
$$\sum_{i\in\mathcal{Q}'} \mathbf{b}_i\bar{\pi}_i + \bar{\mathbf{v}}_{\hat{q}}(\sum_{j\in\mathcal{Q}'}\sum_{l\in\mathcal{L}_{\hat{q}j}} \lambda^{(l)}\mathbf{A}_l) + \sum_{i\in\mathcal{Q}\setminus\hat{\mathcal{Q}}} \bar{\mathbf{v}}_i \sum_{j\in\mathcal{Q}'}\sum_{l\in\mathcal{L}_{ij}} \lambda^{(l)}\mathbf{A}_l$$
$$= \alpha \sum_{i\in\mathcal{Q}'} \bar{\mathbf{v}}_i,$$

where $\mathcal{Q}' = (\mathcal{Q} \setminus \hat{\mathcal{Q}}) \cup \hat{q}$. □

Now we will reduce the SHS model represented in Figure 3.1 by using Theorem 2.

As seen in Table A.1, the discrete states $q_l = \{2, 4\}$ and $q_l = \{0, 1, 3\}$ have the same set of outgoing transitions defined by $(\lambda^{(l)}, \mathbf{x}\mathbf{A}_l)$ tuples, except the $q = 0$ does not have any transition with rate $\lambda^{(l)} = \mu$. According to Proposition 1, adding a self-transition l with $\lambda^{(l)} = \mu$ and $\mathbf{x}\mathbf{A}_l = [x_0 - x_1 \;\; x_2 \;\; 0]$ does not affect the dynamics of the SHS model since it is ensured that in state $q = 0$, continuous states x_1 and x_2 are always zero, and $\mathbf{A}_l$ has the same effect on continuous state as identity reset map. Hence, we can represent the same system with an SHS model with $\mathcal{Q}' = \{0, 1\}$, where $\hat{q} = 0 \in \mathcal{Q}'$ is the compound state of original discrete states $q = \{0, 1, 3\} \subset \mathcal{Q}$ and $\hat{q} = 1 \in \mathcal{Q}'$ is the compound state of original discrete states $q = \{2, 4\} \subset \mathcal{Q}$.

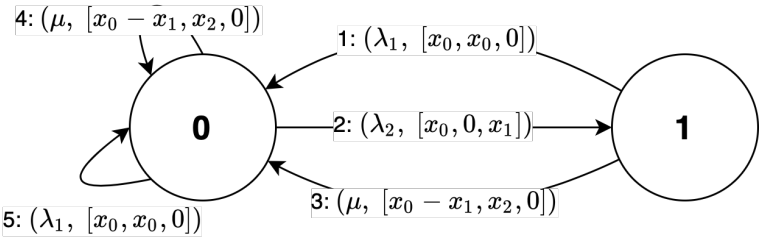

Figure 3.3: A more simplified SHS model for LCFS-Preemptive Scheduling. The detailed information on transitions is given in Table A.3.

The simplified SHS model can be seen in Figure 3.3. As you can see, the self-transition in $q = 1$ is omitted by Proposition 1. Since the SHS model is reduced by the Theorem 2, the discrete states $q \in \{0, 1\}$ are not determined based on descriptive definitions. Yet we can define the discrete states verbally, as follows:

- $(q = 0)$: The server is idle, or serving previously preempted packet, or serving a fresh packet of source 1.
- $(q = 1)$: The server is serving a fresh packet of source 2.

Also, for this model, we refer the reader to the List 3.2.1 for detailed information on transitions.

3.2.3 Average AoI Analysis for LCFS-Preemptive Scheduling

Based on the SHS model represented in Figure 3.3, we obtain the closed-form expression for average AoI for LCFS-Preemptive Scheduling, by solving (3.18).

Theorem 3. *2 sources with offered loads ρ_1, ρ_2 and separate LCFS buffers, scheduled by a server using LCFS-Preemptive Algorithm with service rate μ and a total load $\rho = \sum_{i=1}^{2} \rho_i$, have average ages Δ_1, Δ_2, such that*

$$\Delta_i = \frac{1}{\mu}\left(\frac{1}{\rho_i} + \frac{\rho + 1}{\rho_i + 1}\right) \tag{3.26}$$

Proof. As mentioned in Theorem 1, for solving the SHS model, we need to calculate the steady-state value of test functions denoted as $\bar{\mathbf{v}}_{\bar{q}} = [\bar{v}_{\bar{q}0}, \ldots, \bar{v}_{\bar{q}n}]$ for each $\bar{q} \in \mathcal{Q}$.

With $\mathbf{b}_{\bar{q}} = [1 \quad 0 \quad 0]$, for each $\bar{q} \in \mathcal{Q}$, (3.18) becomes such that

$$(\mu + \lambda)\begin{bmatrix} \bar{v}_{00} \\ \bar{v}_{01} \\ \bar{v}_{02} \end{bmatrix} = \begin{bmatrix} \bar{\pi}_0 \\ 0 \\ 0 \end{bmatrix} + \mu\begin{bmatrix} \bar{v}_{00} - \bar{v}_{01} \\ \bar{v}_{02} \\ 0 \end{bmatrix} + \lambda_1\begin{bmatrix} \bar{v}_{00} \\ \bar{v}_{00} \\ 0 \end{bmatrix} + \mu\begin{bmatrix} \bar{v}_{10} - \bar{v}_{11} \\ \bar{v}_{12} \\ 0 \end{bmatrix} + \lambda_1\begin{bmatrix} \bar{v}_{10} \\ \bar{v}_{10} \\ 0 \end{bmatrix} \tag{3.27a}$$

$$(\mu + \lambda_1)\begin{bmatrix} \bar{v}_{10} \\ \bar{v}_{11} \\ \bar{v}_{12} \end{bmatrix} = \begin{bmatrix} \bar{\pi}_1 \\ 0 \\ 0 \end{bmatrix} + \lambda_2\begin{bmatrix} \bar{v}_{00} \\ 0 \\ \bar{v}_{01} \end{bmatrix} \tag{3.27b}$$

The first thing we will see that $\bar{v}_{02} = \bar{v}_{11} = 0$. By excluding these irrelevant terms we rewrite the equations.

$$(\mu + \lambda)\bar{v}_{00} = \bar{\pi}_0 + \mu(\bar{v}_{00} + \bar{v}_{10} - \bar{v}_{01}) + \lambda_1(\bar{v}_{00} + \bar{v}_{10}) \tag{3.28a}$$

$$(\mu + \lambda)\bar{v}_{01} = \mu\bar{v}_{12} + \lambda_1(\bar{v}_{00} + \bar{v}_{10}) \tag{3.28b}$$

$$(\mu + \lambda_1)\bar{v}_{10} = \bar{\pi}_1 + \lambda_1\bar{v}_{00} \tag{3.28c}$$

$$(\mu + \lambda_1)\bar{v}_{12} = \lambda_2\bar{v}_{01} \tag{3.28d}$$

Adding $\lambda_2 \bar{v}_{10}$ term to both sides of (3.28c) and summing up with (3.28a), and substituting the result in (3.28d) yield

$$\bar{v}_{01} = \frac{1}{\mu}, \qquad \bar{v}_{12} = \frac{\lambda_2}{\mu(\mu + \lambda_1)}, \tag{3.29}$$

since $\bar{\pi}_0 + \bar{\pi}_1 = 1$.

Given that $\Delta = \bar{v}_{00} + \bar{v}_{10}$ by the Theorem 1, substituting (3.29) in (3.28b) results in

$$\Delta = \frac{1}{\mu}\left(\frac{1}{\rho_1} + \frac{\rho + 1}{\rho_1 + 1}\right), \tag{3.30}$$

where $\rho_i = \lambda_i/\mu$ and $\rho = \rho_1 + \rho_2 = \lambda/\mu$.

□

Note that although the existence of the stationary state probabilities of the SHS system is a must for the solution, since $\mathbf{b}_{\bar{q}} = [1 \quad 0 \quad 0]$ for each discrete state $\bar{q} \in \mathcal{Q}$, it is not necessary to compute the stationary probabilities.

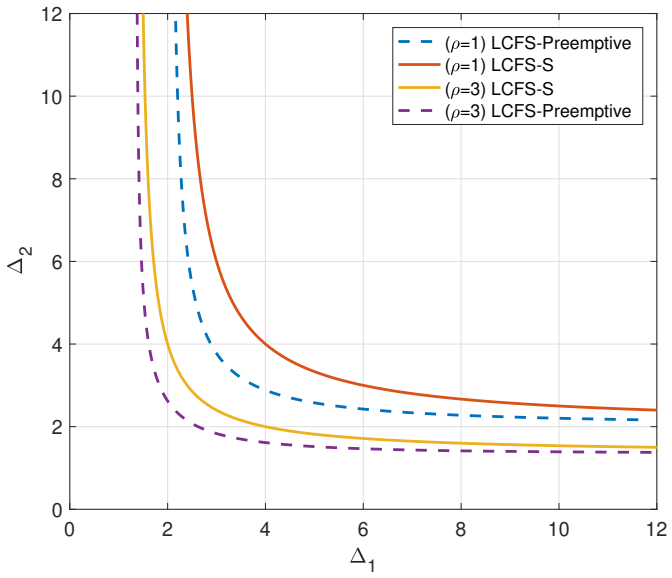

Figure 3.4: Comparison between achievable age regions of LCFS-S and LCFS-Preemptive. Dashed lines and solid lines represent the results of LCFS-Preemptive and LCFS-S, respectively.

In Figure 3.4, we compare LCFS-S with the LCFS-Preemptive Scheduling model. Actually, the only difference between these network models is that in LCFS-Preemptive

Scheduling, each source has its own 1-packet buffer, unlike the common buffer in LCFS-S. With these additional buffers, when a preemption occurs to source 1 by the new arrival of source 2, the preempted packet, which is still age-effective for source 1, can be kept for the next service attempt. Since the preemption directly results in discarding the packet in LCFS-S, the age-effective packets cannot be served when the server is idle. LCFS-Preemptive Scheduling exploits the server's idle duration for sending preempted but still age-effective packet to improve the age performance. In Figure 3.4, the achievable age region of both LCFS-S and LCFS-Preemptive Scheduling is illustrated for the service rate $\mu = 1$. As you can see in the Figure 3.4, additional buffers to store the preempted but still age-effective packets in LCFS-Preemptive Scheduling increase the achievable age region significantly.

In the next subsection, we will introduce the SHS model for LCFS-Nonpreemptive Scheduling defined in Definition 3.2.1.

3.2.4 LCFS-Nonpreemptive Scheduling

Similar to the LCFS-Preemptive Scheduling discussed in subsection 3.2.1, in this model, the sources receive new packets according to independent Poisson processes with rate $\lambda_k, \quad k \in \{1, 2\}$, and each source has its own buffer with a size of 1-packet. The server schedules among the sources according to the rule given in Definition 3.2.1, and serves the selected source according to nonpreemptive service policy, which imposes the next service to wait until the ongoing service's completion.

In the Figure 3.5, the graphical representation of SHS model illustrates LCFS-Nonpreemptive Scheduling can be seen. The discrete states are explained in the list, where each discrete state accompanies with a network case stated by three-digit notation. The first digit indicates the source receiving service; the second digit states the source which will be served next; and the third digit indicates the source will be served in the third place. If any of the digits is zero, it means that there is no packet waiting in that order.

- $q = 0$, (000) : The server is idle.
- $q = 1$, $(100, 200)$: The server is serving either source 1 or source 2, and there

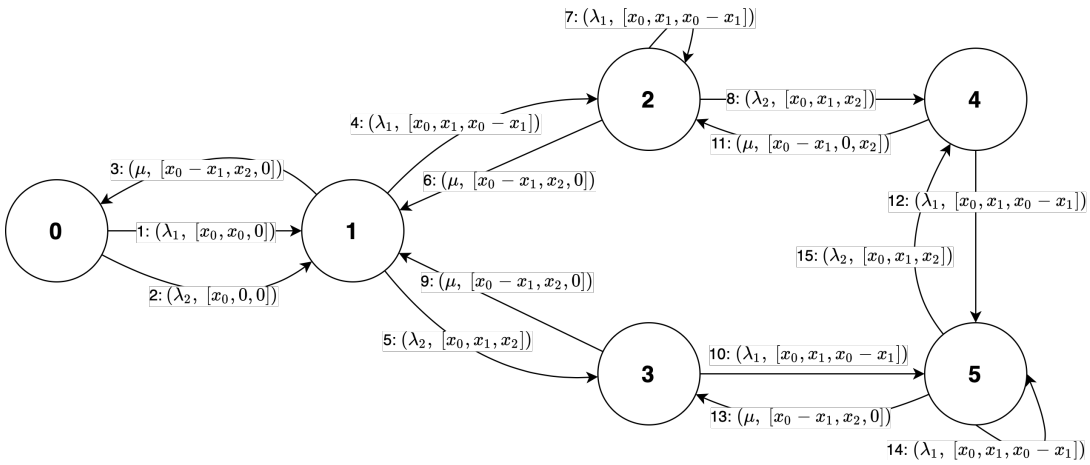

Figure 3.5: SHS model for LCFS-Nonpreemptive Scheduling. The detailed information on transitions is given in Table A.4

isn't any packet waiting in the queues.

- $q = 2$, $(110, 210)$: The server is serving either source 1 or source 2, and there is a packet of source 1 waiting to be served in the next place.
- $q = 3$, $(120, 220)$: The server is serving either source 1 or source 2, and there is a packet of source 2 waiting to be served in the next place.
- $q = 4$, $(121, 221)$: The server is serving either source 1 or source 2, and there is a packet of source 2 waiting to be served in the next place, meanwhile there is a packet of source 1 waiting to be served in the third place.
- $q = 5$, $(112, 212)$: The server is serving either source 1 or source 2, and there is a packet of source 1 waiting to be served in the next place, meanwhile there is a packet of source 2 waiting to be served in the third place.

The information on the transitions of LCFS-Nonpreemptive Scheduling can be found in Table A.4. Still, we explain the events corresponding to the transitions. In LCFS-Nonpreemptive Scheduling, we have five different events:

1st Event, "$(\lambda_1,\ [x_0, x_0, 0])$": This event is for the new arrival of source 1, when the server is idle. It occurs with the exponential rate λ_1. The new packet starts to be served immediately. The continuous state x_1 is set to the current age value, which will

indicate the age drop once the packet is served. This event occurs only in discrete state $q = 0$.

2nd Event, "$(\lambda_1,\ [x_0, x_1, x_0 - x_1])$": This event indicates the new arrival of source 1 in all discrete state, except the idle state $q = 0$. It occurs with the exponential rate λ_1. Since the service policy is nonpreemptive, the new arrival of source 1 waits in the queue until ongoing service is completed. The continuous state x_2 is set to be the difference of the current age and current age drop. In other words, it is the rest of the total age drop of two packets (currently served and waiting), after the ongoing service is completed.

3rd Event, "$(\lambda_2,\ [x_0, x_1, x_2])$": This event stands for the new arrival of source 2. It occurs with the exponential rate λ_2. Since the service policy in this model is nonpreemptive, the new arrival of source 2 does not affect the continuous state $\mathbf{x}(t)$, but it may result in transition to another discrete state. This event occurs in every discrete state $q \in \mathcal{Q}$. Because the ones in discrete states $q = \{3, 4\}$ result in self-transitions with the identity reset map, they are discarded in the Figure 3.5.

4th Event, "$(\mu,\ [x_0 - x_1, x_2, 0])$": This event indicates the completion of the ongoing service. It occurs with the exponential rate μ. When the service is completed, the current age is dropped by the amount of x_1. Additionally, with the service completion, the server starts to serve the waiting packet. Therefore, the continuous state x_1 is set to be x_2, and since there isn't any new arrival, continuous state x_2 is set to be zero. This event occurs when the system is in discrete state $q = \{1, 2, 3, 5\}$.

5th Event, "$(\mu,\ [x_0 - x_1, 0, x_2])$": This event also indicates the completion of an ongoing service, but it only occurs in discrete state $q = 4$. Unlike the 4th Event, since the next source to be served is second one, the service completion reset continuous state x_1 to zero, and continuous state x_2 does not change. Similarly, the service completion causes the continuous state x_0 to the drop by the amount of x_1. This event occurs with the exponential rate μ.

Given the transition/reset maps and transition rates, the close-form expression for an AoI SHS can be computed by Theorem 1. However, if the state space is large, it is

quite challenging to find a solution. Therefore, we use the matrix representation of (3.18) and solve it with the help of MATLAB script we called Generic SHS Solver. With the help of the framework in this script, complex SHS models can be solved easily. Now, we will provide the matrix representation mentioned above.

The first thing we need to calculate is the stationary state probability vector $\bar{\boldsymbol{\pi}}$ of the corresponding continuous-time Markov chain. We define the following matrices

$$\mathbf{D} = diag(d_0, \ \dots \ , d_m), \qquad d_{\bar{q}} = \sum_{l \in \mathcal{L}_{\bar{q}}} \lambda^{(l)}, \tag{3.31a}$$

$$\mathbf{Q} = \begin{bmatrix} a_{00} & \dots & a_{0m} \\ \vdots & \ddots & \vdots \\ a_{m0} & \dots & a_{mm} \end{bmatrix}, \qquad a_{ij} = \sum_{l \in \mathcal{L}_{ij}} \lambda^{(l)}, \tag{3.31b}$$

satisfy $\bar{\boldsymbol{\pi}}\mathbf{D} = \bar{\boldsymbol{\pi}}\mathbf{Q}$ together with the fact that

$$\sum_{i=0}^{m} \bar{\pi}_i = 1$$

where, $\mathcal{L}_{ij} = \{l \in \mathcal{L} : q_l = i, q'_l = j\}, \quad i, j \in \mathcal{Q}$.

Similar to (3.31b), we define a block matrix $\mathbf{R}$ such that block i, j of $\mathbf{R}$ is

$$\mathbf{R}_{ij} = \sum_{l \in \mathcal{L}_{ij}} \lambda^{(l)} \mathbf{A}_l, \qquad i, j \in \mathcal{Q} \tag{3.32}$$

We also define the following block matrices.

$$\mathbf{B} = diag(\mathbf{b}_0, \ \dots \ , \mathbf{b}_m) \tag{3.33a}$$

$$\mathbf{P} = diag(d_0\mathbf{I}_n, \ \dots \ , d_m\mathbf{I}_n) \tag{3.33b}$$

We can rewrite (3.18) in vector form, with these block matrices and a long row vector $\bar{\mathbf{v}} = [\bar{\mathbf{v}}_0 \ \dots \ \bar{\mathbf{v}}_m]$.

$$\bar{\mathbf{v}}\mathbf{P} = \bar{\boldsymbol{\pi}}\mathbf{B} + \bar{\mathbf{v}}\mathbf{R} \tag{3.34}$$

When we solve (3.34) for Figure 3.5 using Generic SHS Solver, the resulting closed-form of average AoI is quite long. Therefore, we refer the reader to our script *Generic SHS Solver* to regenerate the results.

In Figure 3.6, the LCFS-W and LCFS-Nonpreemptive Scheduling are compared. Similarly, the main difference between LCFS-W and LCFS-Nonpreemptive is that

in the LCFS-Nonpreemptive Scheduling model, each source has its own separate 1-packet queue, but in LCFS-W model, the system has a common buffer, and it can store one packet in LCFS manner, in addition to the packet getting served. The waiting room provided in LCFS-W closes the performance gap between LCFS-W and LCFS-Nonpreemptive, but due to the separate buffers in LCFS-Nonpreemptive, in this model, a lower average age for the users is still achievable. The superior performance of LCFS-Nonpreemptive Scheduling model is due to the fact that a packet is not discarded in this model if it is still age-effective for its flow.

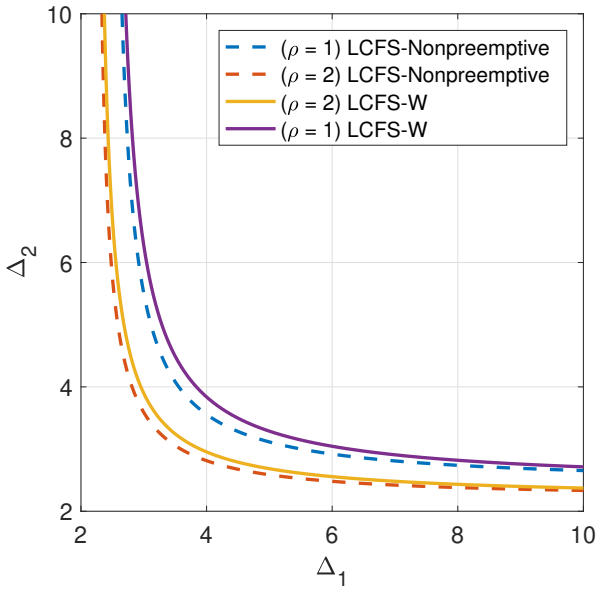

Figure 3.6: Achievable average age regions for LCFS-Nonpreemptive Scheduling and LCFS-W. Dashed lines and solid lines represent the results of LCFS-Nonpreemptive and LCFS-W, respectively.

As you may realize, the network models analyzed with the SHS models have exponential service distribution and Poisson arrivals. In the next subsection, we will discuss a way to use SHS models to analyze hypoexponential type service distributions, which can be specified as the summation of exponential distributed random variables such as Erlang distribution.

3.3 Analyzing Erlang Distributed Service with SHS

In the literature, the SHS approach is used to analyze the network models with exponential characteristics, such as packet arrivals based on a Poisson process and exponentially distributed service times. In this study, we manage to construct SHS models with non-exponential service times by using multi-hop service sequences.

Lemma 1. *Consider a continuous-time Markov chain* $\{X_n; n \geq 0\}$. *Given the CDF of holding time* $\{U_n; n \geq 1\}$ *between state transitions, such that*

$$\Pr\{U_n \leq t | X_{n-1} = j, X_n = k\} = 1 - e^{-\lambda_{jk} t}, \tag{3.35}$$

and the fact that the successive holding intervals $\{U_n; n \geq 1\}$ *are independent of each other, the probability density function of the total duration until nth state transition* S_n *is defined as*

$$f_{S_n|X_0=i}(t) = f_{U_1+U_2+\cdots+U_n|X_0=i}(t), \quad n \geq 1, \tag{3.36}$$

where λ_{jk} *is the exponential transition rate from state* $X_{n-1} = j$ *to* $X_n = k$, *and* $X_0 = i$ *where* i *is any given element of state space.*

In the following, we use the Lemma 1 to analyze a specific case, where the exponential rates of the successive transitions are identical, and a sequence of successive transitions corresponds to a service of the network model.

In an AoI SHS with $(\mathcal{Q}, \mathcal{B}, \mathcal{A})$ representing a network model with Poisson arrivals and exponentially distributed service times, there are transitions $l \in \mathcal{L}$ with $\lambda^{(l)} = \mu$, which correspond to a service completion. Based on the given AoI SHS, we can build a new AoI SHS for a version of the same network model with Erlang distributed service times. To accomplish this, we need to replace the corresponding transitions with a multihop transition sequence for a service completion. Although the resulting SHS can be reduced in some setups, the mentioned change requires including new discrete states with similar outgoing transitions with the origin of the service transition. By using Lemma 1, a k-hop transition sequence with the successive transitions l with $\lambda^{(l)} = \mu$ forms a service with Erlang distribution with the shape parameter k and rate

μ, where the mean service time is

$$\mathrm{E}[S] = \frac{k}{\mu}. \tag{3.37}$$

To explain establishing SHS model with Erlang distributed service distributions, we will provide the versions of LCFS-S, and LCFS-Preemptive and LCFS- Nonpreemptive Scheduling models with the Erlang distributed service times.

Investigating the Erlang distributed service times introduce many additional discrete states and transitions to the model. Additionally, the preemptive and nonpreemptive service policies require different approaches for employing Erlang service distribution. Therefore, we will explain the concept of employing Erlang distribution for preemptive and nonpreemptive service policy in different subsections, beginning with LCFS-S, which has the simplest SHS model among all.

3.3.1 Preemptive Service Policy

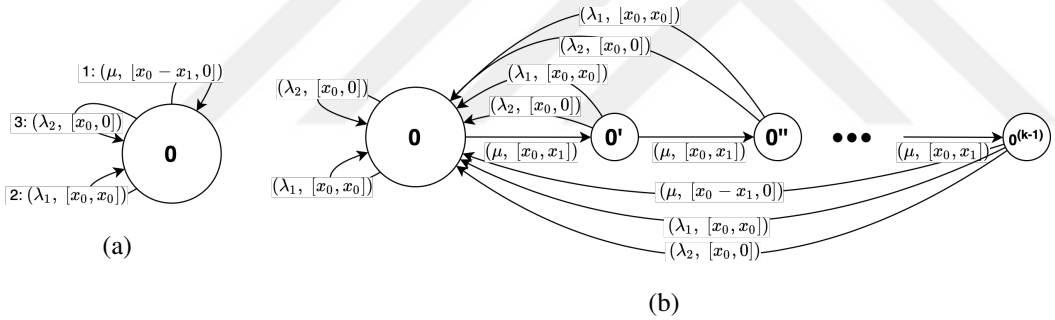

Figure 3.7: SHS models of (a) LCFS-S and (b) its version with Erlang service distribution. The service transition 1 in (a) is represented as a sequence of exponentially rated transitions in (b) to perform Erlang service distribution.

In Figure 3.7a, the LCFS-S is depicted using the SHS framework we used in this work. Although the SHS model is different from the one in [49], both models represent the same network setup. This SHS model is significantly simple; it has a single discrete state and two continuous states x_0 and x_1, representing the age process of interest and age-drop after a successful service, respectively. Additionally, this model has only three transitions. Since the mechanics of this SHS model is same as the previous ones,

we won't explain details of the model. We also refer the reader to [49] for the details. In Figure 3.7b, a version of the LCFS-S with Erlang-k distributed service times can be seen. As you can see, the transition $\{l = 1;\ \lambda^{(l)} = \mu\}$ in Figure 3.7a is replaced in Figure 3.7b with a multi-hop service transition sequence, which includes new discrete states and additional transitions between these states. We denote the discrete states in the multi-hop transition sequence with the same state number together with a prime notation. These new states are illustrated with smaller circles in Figure 3.7b. For a model with Erlang-k distributed service times, we need $k-1$ new discrete states for each transition $\{l :\ \lambda^{(l)} = \mu\}$.

We will denote the discrete states in multi-hop transition sequence as the substates of the origin state of the transition, and we will define a set of substates of discrete state $\bar{q}$ as $q \in \mathcal{Q}_{\bar{q}} \subseteq \mathcal{Q}$. Note that the substates $q \in \mathcal{Q}_{\bar{q}}$ share the same set of transitions $\{l : \lambda^{(l)} = \{\lambda_1, \lambda_2\}\}$ having the same reset map $\mathbf{A}_l$ and the target discrete state q'_l. Additionally, each substate has a transition $\{l :\ \lambda^{(l)} = \mu\}$ which targets next successive state, such that the transitions form a chain of successive transitions having identity reset maps, and the final transition resets the continuous state $\mathbf{x}(t)$ in the same way the transition $\{l :\ \lambda^{(l)} = \mu\}$ of origin discrete state $\bar{q}$ does.

Before providing the closed-form average age solution for LCFS-S with Erlang-k distributed service times, we need the definitions below.

$$V_{\bar{q}j} := \bar{v}_{\bar{q}j} + \sum_{q \in \mathcal{Q}_{\bar{q}}} \bar{v}_{qj}, \qquad j \in \{1, 2, \ldots, n\} \tag{3.38a}$$

$$V_j := \sum_{q \in \mathcal{Q}} \bar{v}_{qj}, \qquad j \in \{1, 2, \ldots, n\}. \tag{3.38b}$$

Note that average age of interest is $\Delta = V_0$.

When the components in (3.18) for LCFS-S with Erlang-k distributed service times 3.7b are reorganized by using the definitions given in (3.38), the set of equations

becomes;

$$V_0(\mu+\lambda) = 1 + (\mu+\lambda)V_0 - \mu\bar{v}_{0^{(k-1)}1}, \tag{3.39a}$$

$$\bar{v}_{01}(\mu+\lambda) = \lambda_1 V_0, \tag{3.39b}$$

$$\bar{v}_{0'1}(\mu+\lambda) = \mu\bar{v}_{01}, \tag{3.39c}$$

$$\bar{v}_{0''1}(\mu+\lambda) = \mu\bar{v}_{0'1}, \tag{3.39d}$$

$$\vdots$$

$$\bar{v}_{0^{(k-1)}1}(\mu+\lambda) = \mu\bar{v}_{0^{(k-2)}1}, \tag{3.39e}$$

where k is the shape parameter of the Erlang distribution. Note that due to the preemptive service policy, we can generalize the relation between the origin state and substates, such that

$$\bar{v}_{q^{(i)}j} = \left(\frac{\mu}{\mu+\lambda}\right)^i \bar{v}_{\bar{q}j}, \tag{3.40}$$

where $q^{(i)} \in \mathcal{Q}_{\bar{q}},\ j \in \{1,\ldots,n\},\ i \in \{1,\ldots,k-1\}$.

Having the set of equations (3.39), we can compute the closed-form expression of average age for this model.

Theorem 4. *Two sources providing Poisson arrivals with rates λ_1, λ_2 to a joint M/Erlang-k/1 LCFS queue with Erlang distribution with rate μ and shape parameter k, and a total arrival rate $\lambda = \lambda_1 + \lambda_2$, have average ages Δ_1, Δ_2, such that*

$$\Delta_i = \frac{1}{\lambda_i}\left(1 + \frac{\lambda}{\mu}\right)^k \tag{3.41}$$

In Figure 3.8, the SHS model of LCFS-Preemptive Scheduling with Erlang-k distributed service times is depicted. Similarly, each transition $\{l : \lambda^{(l)} = \mu\}$ is replaced with a multi-hop transition sequence. Due to the similarities between LCFS-S and LCFS-Preemptive Scheduling, the expected values of test functions $\{\bar{v}_{\bar{q}j} : j \in \{1,\ldots,n\},\ \bar{q} \in \mathcal{Q}\}$ follow similar patterns.

Now, we will provide the components in (3.18) for LCFS-Preemptive Scheduling

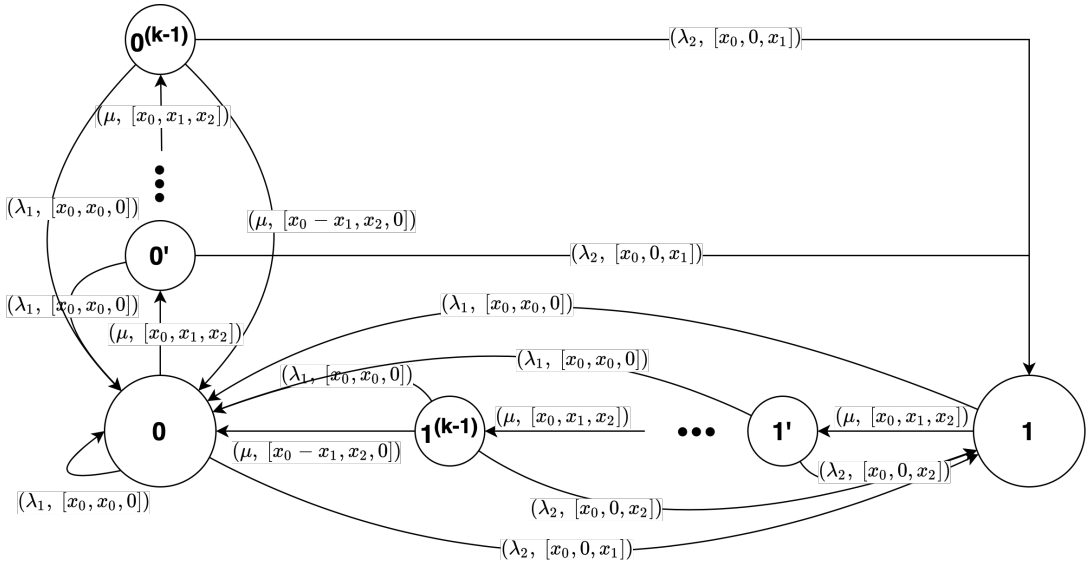

Figure 3.8: SHS model of LCFS-P with Erlang-k distributed service times

with Erlang-k distributed service times.

$$V_0(\mu+\lambda) = 1 + (\mu+\lambda)V_0 - \mu(v_{0^{(k-1)}1}), \tag{3.42a}$$

$$v_{01}(\mu+\lambda) = \lambda_1 V_0 + \mu(v_{1^{(k-1)}2}), \tag{3.42b}$$

$$v_{12}(\mu+\lambda) = \lambda_2(V_{12} + V_{01}), \tag{3.42c}$$

together with (3.40), and

$$V_{\bar{q}j} = \left(1 - \left(\frac{\mu}{\mu+\lambda}\right)^k\right)\frac{\lambda+\mu}{\lambda}v_{\bar{q}j}, \quad j \in \{1,\ldots,n\} \tag{3.42d}$$

complete the set of relevant equations.

By simple algebraic manipulation, we can compute closed-form expression for the average age of LCFS-Preemptive Scheduling with Erlang-k distributed service times.

Theorem 5. *Two sources receiving Poisson arrivals with rates λ_1, λ_2 to separate LCFS queue, scheduled by LCFS-Preemptive Algorithm with the service distribution Erlang(k,λ), and a total arrival rate $\lambda = \lambda_1 + \lambda_2$, have average ages Δ_1, Δ_2, such that*

$$\Delta_i = \frac{1}{\lambda_i}\left[\left(\frac{\mu+\lambda}{\mu}\right)^k - \frac{\beta}{\mu+\lambda-\beta}\right], \tag{3.43}$$

where

$$\beta = \lambda_{-i}\left[\frac{\mu}{\lambda}\left[1-\left(\frac{\mu}{\mu+\lambda}\right)^{k-1}\right]+1\right], \quad \lambda_{-i} = \lambda - \lambda_i. \tag{3.44}$$

3.3.2 Nonpreemptive Service Policy

As discussed in 3.3.1, due to the preemptive policy, upon a new arrival, the system preempts the ongoing service and starts a new one. Therefore, generating an SHS model for preemptive policies are relatively straightforward, also for the case with Erlang distributed service times. However, the case with the nonpreemptive service policy requires keeping the current situation of the ongoing service, even if a new packet arrives. Therefore, the SHS model for a network with nonpreemptive service policy has layers representing the current state of the ongoing service.

We will explain the method of employing Erlang distribution on a network with nonpreemptive service policy on LCFS-W, which is the simplest SHS model for a network with nonpreemptive service policy available.

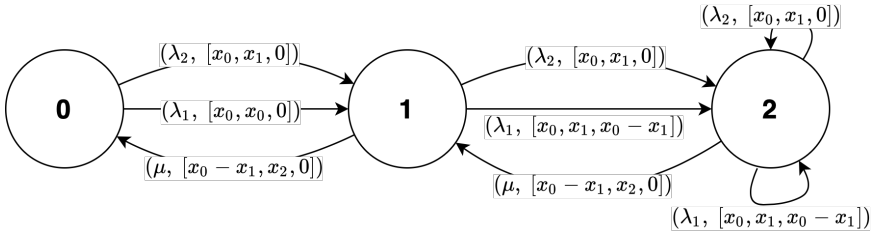

Figure 3.9: SHS model of LCFS-W with exponential service times

In Figure 3.9, the SHS model of LCFS-W is shown. We refer the reader to [49] for the details of the model. As seen in the model, there are two transition $\{l : \lambda^{(l)} = \mu\}$ and we will replace these transitions with the multi-hop transition sequence. Although in the preemptive models the transitions $\{l : \lambda^{(l)} = \{\lambda_1, \lambda_2\}\}$ of the origin states $\{\bar{q}_l : \lambda^{(l)} = \mu\}$ are dublicated in the substates $q \in \mathcal{Q}_{\bar{q}}$, in the nonpreemptive models, the transitions between the states of the origin model happen between substates within a layer, identifying the current situation of the ongoing service.

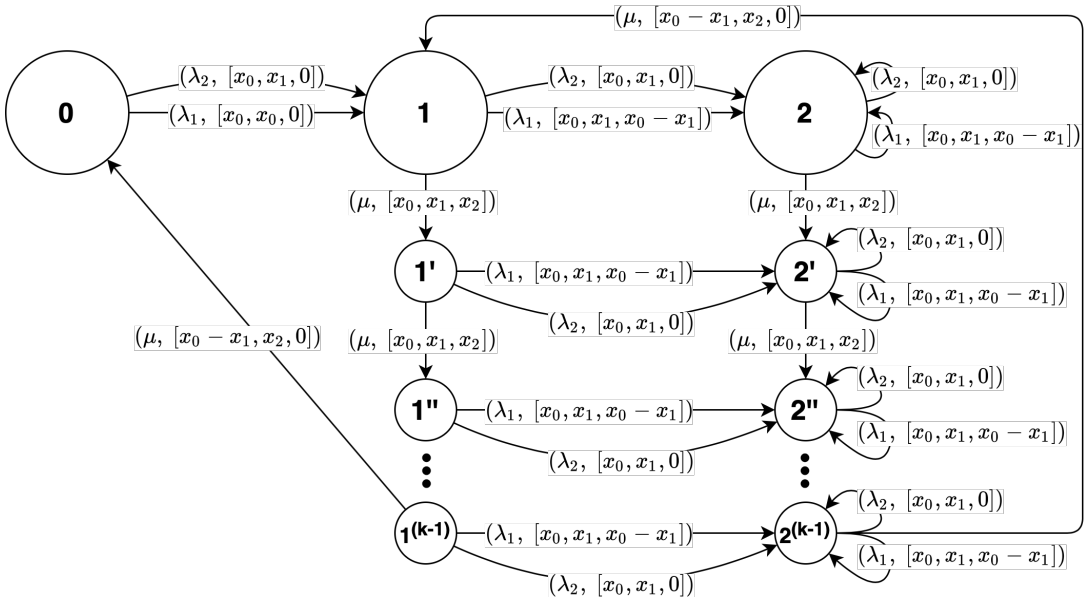

Figure 3.10: SHS model of LCFS-W with Erlang-k service distribution

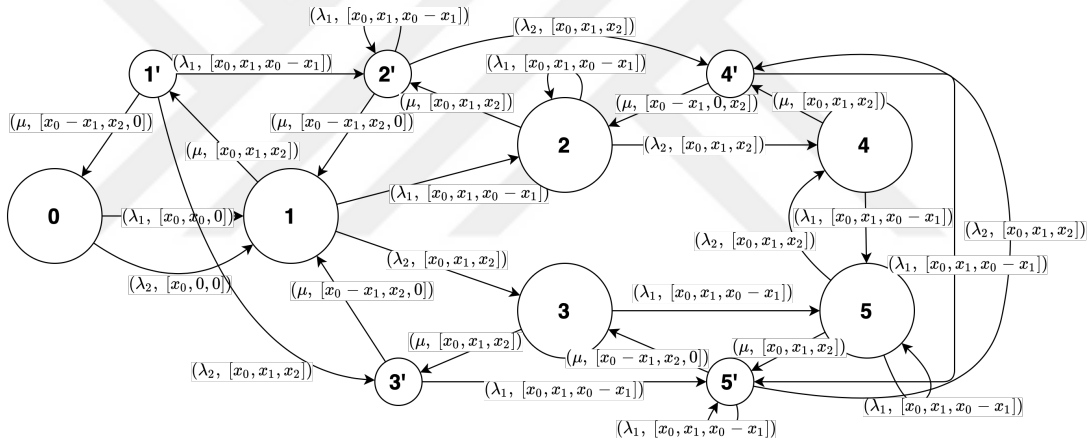

Figure 3.11: SHS model of LCFS-NP with Erlang-2 service distribution

In Figure 3.10, the SHS model of LCFS-W with Erlang-k distributed service times can be seen. Since the only states having a service transition $\{l : \lambda^{(l)} = \mu\}$ are $q \in \{1, 2\}$, the layers of service includes those substates. The transitions corresponding to the new packet arrivals are kept same to occur between the substates within a layer. The system goes to the deeper layer by the transitions having $(\lambda^{(l)} = \mu, \mathbf{A}_l = \mathrm{I})$. At the last layer, since the ongoing service has a single hop to complete, the service transition $\{l : \lambda^{(l)} = \mu\}$ has the same reset map $\mathbf{A}_l$ with the origin model's service transitions.

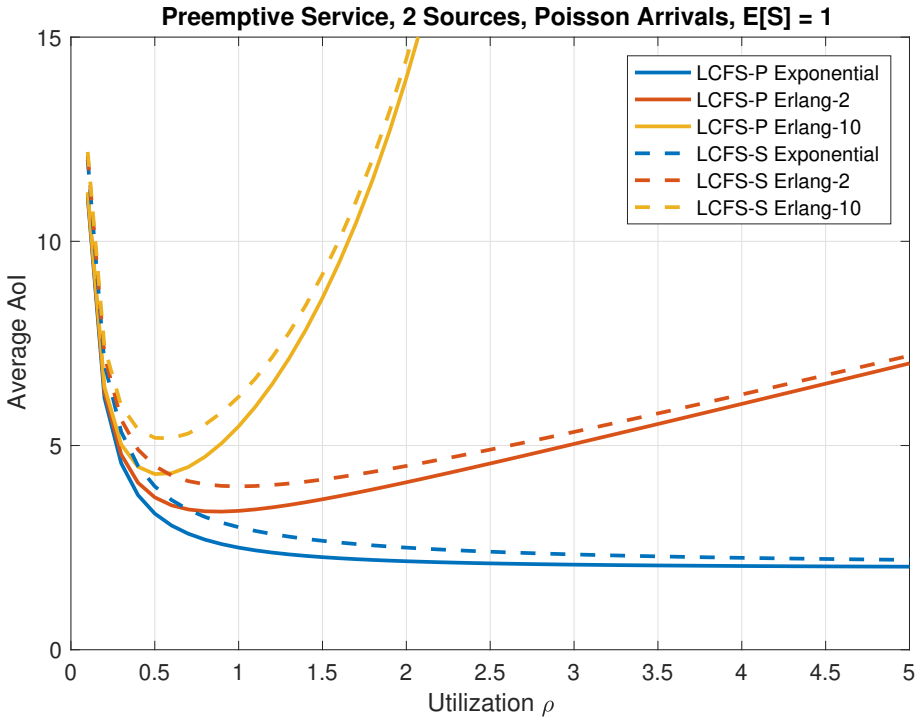

Figure 3.12: Average age versus utilization for the cases of LCFS-Preemptive and LCFS-S under Erlang service distribution with shape parameter k 1, 2 and 10. The dashed lines and solid lines represent LCFS-S and LCFS-Preemptive, respectively.

Similarly, the LCFS-Nonpreemptive Scheduling can be modeled with Erlang distributed service times. In Figure 3.11, we provide the model for Erlang-2 distributed service time. Note that the version with Erlang-k distributed service times can be modeled with the layer-wise structure explained above.

Since in the nonpreemptive models, the system does not return to the first layer at every new arrival, computing a closed-form expression for average age is not an easy task in these models. Therefore, we refer the reader to Generic SHS Solver for the symbolic closed-form expressions for average age.

3.3.3 Numerical Results

In this subsection, we will provide the numerical results for network models with Erlang service distribution and Poisson packet arrivals. The results are generated by

using the closed-form expressions given in subsection 3.3.1 and obtained by Generic SHS Solver. The numerical results include LCFS-S and LCFS-W models proposed in [49], and LCFS-Preemptive and LCFS-Nonpreemptive proposed in this chapter.

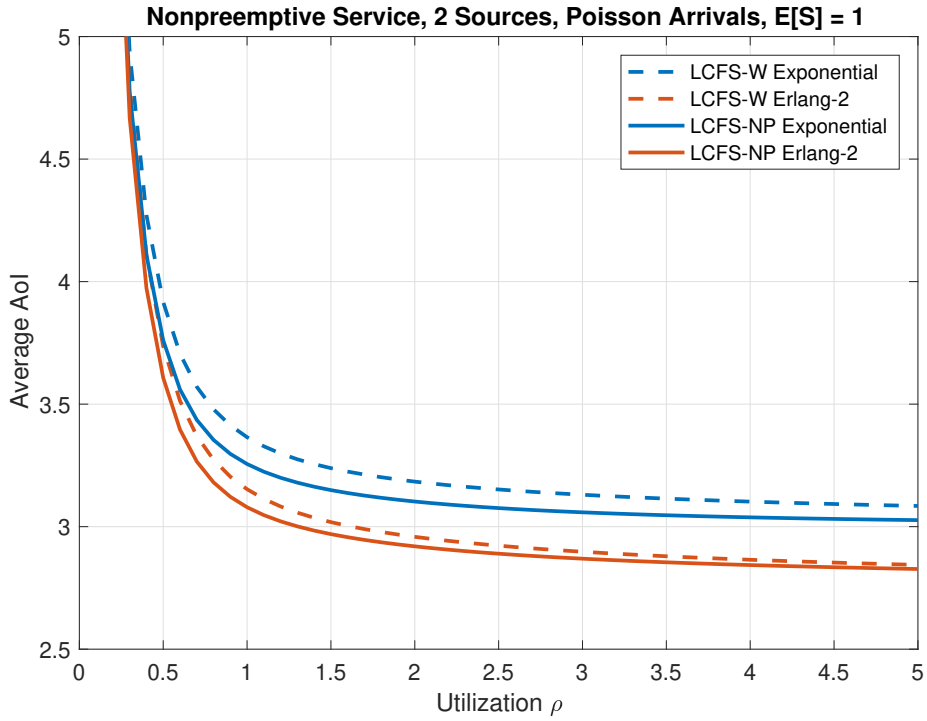

Figure 3.13: Average age versus utilization for the cases of LCFS-Nonpreemptive and LCFS-W under Erlang service distribution with shape parameter k 1 and 2. The dashed lines and solid lines represent LCFS-W and LCFS-Nonpreemptive, respectively.

In Figure 3.12, the average age performance of LCFS-Preemptive and LCFS-S are compared under Erlang service distributions with different shape parameters. The dashed lines belong to the results of LCFS-S and straight lines are representing LCFS-Preemptive. These results are produced for two sources having the same Poisson arrival rate. The expected service time $\mathrm{E}[S]$ is kept constant and equals to 1. Similar to the result in Figure 3.4, the additional buffer size provided in LCFS-Preemptive for preempted age-effective packets affects the average age performance positively for Erlang service distribution. The next important point to mention is the negative effect of preemption on models with Erlang service distribution. As seen in Figure 3.12, at high utility, the average age increases with service time distribution is a *New-Better-*

Than-Used type distribution. This effect is discussed for single-source network in [14]. According to the results, this phenomenon also occurs in scheduling scenarios. In Chapter 4, we discuss this effect in detail and propose a novel approach to avoid it.

In Figure 3.13, LCFS-Nonpreemptive and LCFS-W is compared under the same settings used in Figure 3.12 except the service is nonpreemptive. The results in this figure are consistent with Figure 3.6. According to them, we can see that although LCFS-W has a common waiting room with a buffer size of 1-packet, employing separate waiting rooms for each source in LCFS-Nonpreemptive decreases the resulting average age. Additionally, under nonpreemptive service discipline, the Erlang service distribution yields lower average age than the exponential service distribution.

CHAPTER 4

AGE-AWARE SCHEDULING

Preemptive and nonpreemptive LCFS policies have been studied in many works, such as [9, 14, 25, 52–54]. [53] investigated the effect of replications on multi-server networks. The Age of Information on multi-flow networks are discussed in [25, 35, 55]. [22, 25] investigated a similar scheduling algorithm, called Maximum Age First (MAF), which can be used in multi-flow networks. In this paper, we adopt a model similar to the one in [25], and proposed our scheduling policy called Maximum Age First. The optimality of MAF policy is proved in [37] under preemptive service discipline for synchronous sources, which means they receive the packet arrivals at the same time instants. However, in the model studied in this work, the case with independent packet arrivals is considered.

In section 4.1, the details of the system models used throughout the chapter are introduced. Then, in section 4.2, we defined the Maximum Age Difference policy. The MAD policy is examined for preemptive and nonpreemptive service disciplines separately. In section 4.3, the optimality of MAD policy for general service and arrival processes is proved under nonpreemptive service discipline. Before examining the MAD policy under preemptive service, section 4.4, we first investigated the effect of preemption on average age in detail, starting with a single-flow network. In 4.4.1, we proved a novel preemption policy called Opportunistic Preemption with is quite useful, especially the models with *New-Better-Than-Used* type service distribution. After that, in 4.4.2 we proposed a preemptive scheduling policy called *Maximum Age Difference - Opportunistic Preemption*. In the end, in 4.5, we provided detailed simulation results comparing the average age performance of MAD and MAF policies.

4.1 System Model

The system we are investigating in this chapter can be seen in Figure 4.1. We consider a model with N independent sources served by a single server. Each status update flow has separate destinations. Therefore, we denote age process of flow i as $\Delta_i(t),\ i \in \{1, \ldots, N\}$. The source i generates new packets according to i.i.d. Poisson process with rate λ_i, and an arriving packet waits in the buffer of the source until it is served. The buffer may work according to LCFS or FCFS discipline. Additionally, we assume the packets generated at sources arrive at the server instantaneously, without any delay. We consider the server is work-conserving, and it is able to serve only one flow at a time. When the server is available, it schedules a flow among all and serves it. During Chapter 4, the service duration is modeled to be either exponentially distributed with the rate μ or Erlang distributed.

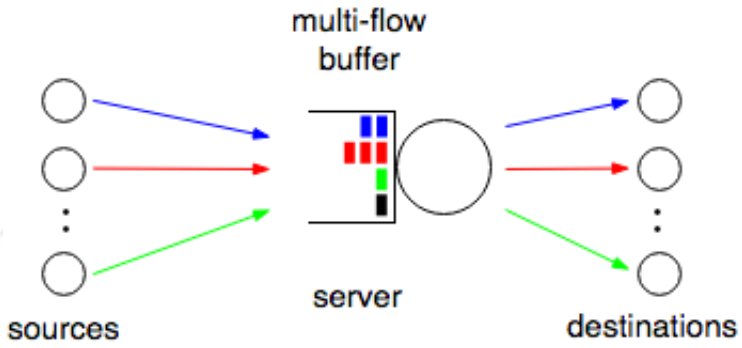

Figure 4.1: Multi-flow network model with separate queues

It is important to mention that, unlike the previous studies, where the arrivals are restricted to be synchronous [25], or occur based on the decision of an optimized sampler [26], the Poisson processes of sources in this work are assumed to be independent of each other and the scheduling decisions.

This model can be thought as a part of a sensor network, the simple sensor units are connected to control unit (server) via short-range wireless links (or wired connections), and the local access point/control unit is responsible for sending these status update packets to remote destinations for monitoring or decision making.

We assume that packets arriving at the server are separately queued for each flow. At

any time instant t, the largest among the time-stamps of the packets received thus far by the destination of the i^{th} flow is $U_i(t)$. The age of information of the corresponding flow is defined as $\Delta_i(t) = t - U_i(t)$. This indicates the freshness of information at each destination node. We define the average instantaneous age of system as

$$\Delta(t) = \frac{1}{N}\sum_{k=1}^{N}\Delta_i(t), \quad t \geq 0, \tag{4.1}$$

and the time average age of information of the system is

$$\bar{\Delta} = \lim_{T\to\infty}\frac{1}{T}\int_0^T \Delta(t)dt \tag{4.2}$$

In the following, we will propose a scheduling algorithm called Maximum Age Difference (MAD) minimizing the average age of information of the system defined in (4.2).

4.2 Maximum Age Difference Algorithm

In MAD, the server schedules the flow with the highest *age difference*, defined as follows. At any time t, the time-stamp of the head-of-line packet belonging to i^{th} flow generated at the source is $V_i(t)$, and the time-stamp of the freshest packet belonging to i^{th} flow received by the destination is $U_i(t)$. The age difference of the i^{th} flow is defined as

$$d_i(t) \coloneqq \max(V_i(t) - U_i(t), 0), \quad i \in \{1, \ldots, N\}. \tag{4.3}$$

Definition 4.2.1. *The Maximum Age Difference (MAD) policy is a work-conserving scheduling algorithm, which decides the next flow to be served according to the criteria*

$$i^*(t) = \arg\max_i d_i(t), \quad i \in \{1, \ldots, N\}, \tag{4.4}$$

with ties broken arbitrarily.

Because of the linear increase of the age process with the slope of 1, except the discontinuous drops, the *age difference* defined in (4.3) equals to the amount by which age process would drop if this flow was scheduled at time t. Therefore, MAD policy

is a greedy policy, maximizes the reduction in the system's age of information by considering the information available at decision instant. Intuitively, if the decisions made by the current reward remain the best decision in the future, the greedy policy is optimal.

The effect of preemption in age performance on LCFS queues is studied in [14] under different service time distributions. Similarly, the preemption in service affects the age performance also for scheduling scenarios. Therefore, we need to examine the MAD policy for the cases when preemption is allowed and preemption is not allowed. In the following section, we will discuss the optimality of MAD policy under nonpreemptive service discipline.

4.3 Nonpreemptive MAD Policy

To prove the optimality of the MAD policy, we will use a similar sample-path technique to the one used in [25] to prove the optimality of MAF policy for synchronized packet generation and arrival processes. Before continuing, we need the following definitions.

Definition 4.3.1. ***Univariate Stochastic Ordering:*** *[56] Given two random variables X and Y, if*

$$\mathrm{P}\{X > \alpha\} \leq \mathrm{P}\{Y > \alpha\}, \quad \forall \alpha \in \mathbb{R},$$

then X is said to be stochastically smaller than Y, denoted as $X \leq_{st} Y$.

Definition 4.3.2. ***Multivariate Stochastic Ordering:*** *[56] Given two random vectors $\mathbf{X}$ and $\mathbf{Y}$, if*

$$\mathrm{P}\{\mathbf{X} \in \mathcal{U}\} \leq \mathrm{P}\{\mathbf{X} \in \mathcal{U}\}, \quad \textit{for all upper sets } \mathcal{U} \subseteq \mathbb{R}^n,$$

then $\mathbf{X}$ is said to be stochastically smaller than $\mathbf{Y}$, denoted as $\mathbf{X} \leq_{st} \mathbf{Y}$.

Definition 4.3.3. ***Stochastic Ordering of Stochastic Processes:*** *[56] Given two stochastic processes $\{X(t), t \in [0, \infty]\}$ and $\{Y(t), t \in [0, \infty]\}$, if the multivariate stochastic ordering holds for any $t_i \in [0, \infty]$, where $t_1 < t_2 < \cdots < t_n$, that*

$$(X(t_1), X(t_2), \ldots, X(t_n)) \leq_{st} (Y(t_1), Y(t_2), \ldots, Y(t_n)), \tag{4.5}$$

then $\{X(t), t \in [0,\infty]\}$ *is said to be stochastically smaller than* $\{Y(t), t \in [0,\infty]\}$*, denoted by* $\{X(t), t \in [0,\infty]\} \leq_{st} \{Y(t), t \in [0,\infty]\}$.

The optimality of nonpreemptive MAD policy is established in the following theorem.

Theorem 6. *For any i.i.d service time distribution, the MAD policy is the optimal among all* $\pi \in \Pi$*, and it follows that*

$$\Delta_{MAD}(t) \leq_{st} \Delta_{\pi}(t), \quad \textit{for } t \geq 0,\ \pi \in \Pi, \tag{4.6}$$

where $\leq_{st}$ *is the stochastic ordering defined in [56], and* Π *is the set of causal, work-conserving and nonpreemptive scheduling policies.*

Consequently, the Theorem 6 results in

$$\mathrm{E}[\bar{\Delta}_{MAD}] \leq \mathrm{E}[\bar{\Delta}_{\pi}], \quad \pi \in \Pi. \tag{4.7}$$

Proof. The main idea of the proof is showing that for the nonpreemptive service discipline, scheduling the flow resulting in the maximum age reduction is also the best choice after the service completion.

Let average instantaneous age of the system under the scheduling policy $\pi \in \Pi$, which is the set of causal, work-conserving and nonpreemptive scheduling policies, be denoted as $\Delta_{\pi}(t) = \frac{1}{N}\sum_{i=1}^{N} \Delta_{i,\pi}(t)$, and let $\{\Delta_{\pi}(t), t \geq 0\}$ denote the state process of the scheduler π. Throughout the proof, we assume that $\Delta_{\pi}(0^{-}) = \Delta_{MAD}(0^{-}) = 0$. We use a coupling argument and forward induction to prove Theorem 6.

Suppose that for any scheduler π, the state processes $\widetilde{\Delta}_{MAD}(t)$ and $\widetilde{\Delta}_{\pi}(t)$ have the same probability distribution with state processes $\Delta_{MAD}(t)$ and $\Delta_{\pi}(t)$. The state processes $\widetilde{\Delta}_{MAD}(t)$ and $\widetilde{\Delta}_{\pi}(t)$ are coupled such that the packet service times and packet generation times are equal under both scheduling policies. Such a coupling is valid since the service time distribution is fixed under all policies. According to Theorem 6.B.30 in [56], if we can show

$$\mathrm{P}\left[\widetilde{\Delta}_{MAD}(t) \leq \widetilde{\Delta}_{\pi}(t), t \geq 0\right] = 1, \tag{4.8}$$

then Theorem 6 is proven.

In the following lemma, we will compare MAD scheduler and scheduler π on a sample path to show (4.8) is correct.

Lemma 2. *(Inductive Comparison). Given that* $\widetilde{\Delta}_{\pi}(0^{-}) = \widetilde{\Delta}_{MAD}(0^{-}) = 0$*, suppose that a packet is delivered under the MAD scheduler and the scheduler* π *at the same time* t_s*. Just before the delivery of the packet, the state processes of the MAD scheduler and scheduler* π *are* $\widetilde{\Delta}_{MAD}(t_s)$ *and* $\widetilde{\Delta}_{\pi}(t_s)$*, which become* $\widetilde{\Delta}_{MAD}(t_s^{'})$ *and* $\widetilde{\Delta}_{\pi}(t_s^{'})$ *just after the delivery. If*

$$\widetilde{\Delta}_{MAD}(t_s) \leq \widetilde{\Delta}_{\pi}(t_s), \tag{4.9}$$

then

$$\widetilde{\Delta}_{MAD}(t_s^{'}) \leq \widetilde{\Delta}_{\pi}(t_s^{'}). \tag{4.10}$$

The proof of Lemma 2 is as follows:

Proof. Since only one source can be scheduled at a time, and the delivered packet reduces the age of its flow by its *age difference*, it also reduces the age of the system by one-Nth of the delivered packet's *age difference*. Noticing that the scheduling decision is taken by comparing the *age differences* at the decision instant t_d, the system states under the MAD scheduler and scheduler π evolve as follows

$$\begin{aligned} \widetilde{\Delta}_{\pi}(t_s^{'}) &= \widetilde{\Delta}_{\pi}(t_s) - \frac{d_i(t_d)}{N} \\ &\geq \widetilde{\Delta}_{MAD}(t_s) - \frac{d_{i^*}(t_d)}{N} \\ &= \widetilde{\Delta}_{MAD}(t_s^{'}), \quad t_s \geq t_d. \end{aligned} \tag{4.11}$$

Since the MAD scheduler ensures that $d_{i^*}(t_d) = \max_i d_i(t_d)$, and $d_{i^*}(t_s) = d_{i^*}(t_d)$ for all causal, work-conserving and nonpreemptive scheduling policies, (4.9) and (4.11) complete the proof of Lemma 2. □

Now, we investigate the evolution of the state processes $\widetilde{\Delta}_{\pi}(t)$ and $\widetilde{\Delta}_{MAD}(t)$ in two cases:

Case 1: When there is no packet delivery, the age of the system grows linearly with a slope of 1.

Case 2: When a packet is delivered, the age of the system evolves according to Lemma 2.

Note that since the arrival processes and service times are fixed for both state processes, both policies start a new service always at the same time, which occur either just after the completion of an ongoing service, or the arrival of a packet to an idle system. This statement is true if $\pi \in \Pi$ is a work-conserving policy.

By induction over time, we obtain

$$\widetilde{\Delta}_{\pi}(t) \geq \widetilde{\Delta}_{MAD}(t), \quad t \geq 0, \tag{4.12}$$

which yields (4.8), and by Theorem 6.B.30 of [56], (4.8) completes the proof. □

Note that the proof of Theorem 6 is not restricted to the LCFS queue discipline. Therefore, nonpreemptive MAD policy is also an age-optimal nonpreemptive scheduling policy for networks with FCFS queues.

4.4 Discussion on Preemptive Policy

A preemptive scheduling policy should define not only the way the scheduling decisions are made but also in which conditions the preemptions occur. Usually, preemptive scheduling policies in the literature, such as [5, 25] always preempt the ongoing service if an arrival occurs and the resulting policies fail, especially when the service time distribution is a New-Better-than-Used (NBU) type distribution, like Gamma distribution. A New-Better-than-Used type distribution is defined as follows:

Definition 4.4.1. *New-Better-than-Used Distributions: Consider a non-negative random variable X with complementary cumulative distribution function (CCDF) $\bar{F}(x) = \Pr[X > x]$. Then, X is said to be New-Better-than-Used (NBU), if for all $t, \tau \geq 0$*

$$\bar{F}(\tau + t) \leq \bar{F}(\tau)\bar{F}(t). \tag{4.13}$$

The exponential distribution is a special case, where (4.13) becomes an equality [57]. Erlang distribution and geometric distribution are two examples of NBU type distributions. In the literature, few works have proposed novel preemption policies [43,58]. The model discussed in [43] uses the AoI-SHS framework to analyze the case with multiple sources which are only able to preempt their own packet. [58] proposes a novel probabilistic preemption policy for a $M/PH/1/1$ queue, which receives arrivals from multiple sources.

Before continuing on the preemptive scheduling policy, we will investigate the preemption policy for a single-flow network in the next subsection. The results of this subsection will be used to propose a better preemptive scheduling policy.

4.4.1 Preemption on a Single-Flow Network

In the network models with an NBU type service distribution and Poisson packet arrivals, even though LCFS queuing discipline is employed, the average age of information vs. arrival rate curve has a *bathtub shape*: In other words, the average age of information decreases at low throughput, and increases at high throughput as throughput increases. This phenomenon is studied in [14] for gamma distributed service times on a single-flow network, where preemptive LCFS queuing discipline is utilized. We discussed a similar phenomenon in Section 3.3, where a basic scheduling scenario is investigated under Erlang distributed service times.

The main cause of this phenomenon is that the regular preemption rule preempts the ongoing service at every new packet arrival. However, preemption is not always beneficial in terms of age of information, even if the new arriving packet's *age difference* is greater than the old one. This can be explained by a simple example, that the service times and arrival times are deterministic and the new packet always arrives just before the ongoing service is completed. In this example, the regular preemption policy never completes the ongoing service; therefore, the age of information grows unboundedly.

In the following, to understand the effect of the preemption on average age, we will examine two scenarios, that in the first one, the new arrival preempts the ongoing

service, and in the second one, the new arrival does not preempt the ongoing service. We assume that the preemption policy used in both scenarios are same except the decision upon the first packet arrival. Additionally, we will keep future arrival and service times fixed, so that we can focus on the effect of a single preemption decision. By comparing the outcomes of these scenarios, we will identify the criteria required for a preemption decision.

Let the area under the instantaneous age of the source until time t under the preemption policy $\phi \in \Pi$ is denoted as

$$g_\phi(t) = \int_0^t \Delta_\phi(\tau) d\tau,$$

and let $\{g_\phi(t), t \geq 0\}$ denote the state process of the system. Suppose that the state process of the first scenario, that the ongoing service of the n^{th} packet is preempted, is denoted by $g_P(t)$, and the state process of the second scenario, that the ongoing service of the n^{th} packet is not preempted, is denoted by $g_{NP}(t)$. These state processes are coupled in such a way that the arrival processes and service times are identical. Additionally, these policies make the same preemption decision for every packet following n^{th} packet. The arrival of n^{th} packet occurs at time s_n, and the packet is delivered at r_n under policy P and at $\bar{r}_n$ under policy NP. Additionally, S_n denotes the service duration of n^{th} packet, $R_{S_n}(\tau_n)$ denotes the residual service time depending on the time passed since service beginning, τ_n, and d_n is the *age difference* of the n^{th} packet. The list of the key notation can be seen in Table 4.1.

Suppose that at $(n+1)^{th}$ packet arrives during the service of n^{th} packet, and due to the coupling argument $\Delta_P(t) = \Delta_{NP}(t)$ for $t \leq t_d$, where $t_d = s_{n+1}$ is the instant, when the preemption decision is made for n^{th} packet. Now, we will compare the age processes when n^{th} packet is preempted and not preempted, illustrated by $\Delta_P(t)$ and $\Delta_{NP}(t)$. The other arrivals are treated according to a fixed preemption policy. We will compare the state processes until both policies serve the same packet together. We denote this instant as t_s, and it satisfies that $\Delta_P(t) = \Delta_{NP}(t)$ for $t \geq t_s$.

In the case in Figure 4.2, there isn't any arrival before the delivery of the $(n+1)^{th}$ packet, i.e. $s_{n+2} \geq \bar{r}_{n+1}$. In this case, the preemption cost $\mathbf{C}_n$ defined by the difference between the state processes becomes

$$\mathbf{C}_n \coloneqq g_P(t) - g_{NP}(t) = A - B, \quad t \geq t_s, \tag{4.14}$$

N	total number of packet, packet index is $n \in [0, N]$
X_n	the interarrival time between $n-1^{th}$ and n^{th} packets
S_n	the service duration of packet n
R_{S_n}	the residual time for random variable S_n
τ_n	the time passed since n^{th} service has begun.
Q_n	$S_n - R_{S_{n-1}}$
π_{NP}	nonpreemptive policy
π_P	preemptive policy
s_n	arrival time of packet n
r_n	delivery time of packet n under P policy
$\bar{r}_n$	delivery time of packet n under NP policy
d_n	age difference of packet n
t_d	preemption decision instant
t_s	time that the same packet is delivered under both policies P and NP
ϕ_n	indicator function that defines the preemption policy
Φ_n	probability that packet n is preempted

Table 4.1: The List of the Key Notations

where $A = d_n(X_{n+1})$ and $B = (d_{n+1} - d_n)R_{S_n}$, and $Q_{n+1} = S_{n+1} - R_{S_n}$. It means that if $\mathbf{C}_n > 0$, then not preempting the n^{th} packet is advantageous in terms of AoI, or vice versa. Note that this case has a high probability, when the network operates at a low utility point, i.e. $\mathrm{E}[S] \ll \mathrm{E}[X]$, where $\mathrm{E}[S]$ and $\mathrm{E}[X]$ are expected service and interarrival times.

However, it is also possible that a packet arrives prior to $\bar{r}_{n+1}$. Therefore, we need to consider other cases when calculating the exact expected cost of preemption. In the cases shown in Figure 4.3, the arrival of $(n+2)^{th}$ packet between $\bar{r}_n$ and $\bar{r}_{n+1}$ preempts $(n+1)^{th}$ packet. At the left figure, $(n+2)^{th}$ packet arrives between $\bar{r}_n$ and r_{n+1}, and at right, $(n+2)^{th}$ packet arrives between r_{n+1} and $\bar{r}_{n+1}$. Remember, $(n+2)^{th}$ packet may also be preempted if another packet arrives before its service is completed. Note that we consider these cases to make the decision for the preemption of n^{th} packet. Therefore, the optimal preemption decision is dependent on the

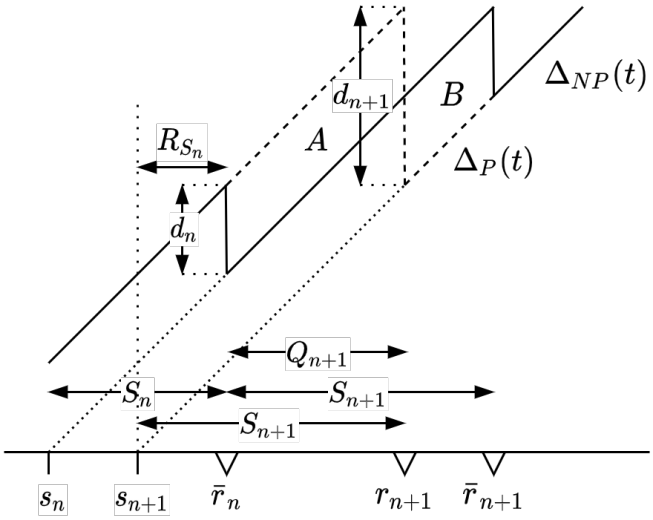

Figure 4.2: $\Delta_P(t)$ and $\Delta_{NP}(t)$ when there isn't any arrival before $\bar{r}_{n+1}$, that preempts $(n+1)^{th}$ packet.

service and arrival times of future packets until the earliest joint service completion. Under these circumstances, the best approach for preemption problem is optimizing the preemption decisions in expectation.

Given the i.i.d service and arrival times,we define a causal work-conserving preemption policy ϕ that depends on the age differences of current and new arrived packets and the service time has already passed.

$$\phi_n(d_n, d_{n+1}, \tau_n) = \begin{cases} 1, & \text{if } n^{th} \text{ packet is preempted} \\ 0, & \text{if } n^{th} \text{ packet is not preempted} \end{cases} \tag{4.15}$$

where τ_n is the time passed since service of n^{th} packet begins, and d_n is the age difference of n^{th} packet. Note that $\mathrm{E}[\phi_n]$ gives us the probability that the n^{th} packet is preempted, which is shortly denoted by Φ_n.

If we consider all possible arrivals and preemption decisions, the preemption cost for n^{th} packet (4.14) becomes

$$\mathbf{C}_n^{\phi} \coloneqq g_P(t_s) - g_{NP}(t_s) = \mathbf{A}_n(d_n, d_{n+1}, \tau_n) - \mathbf{B}_n(d_n, d_{n+1}, \tau_n), \tag{4.16}$$

where $\mathbf{A}_n$ and $\mathbf{B}_n$ are two random variables that equal to the upper and lower area between Δ_{NP} and Δ_P.

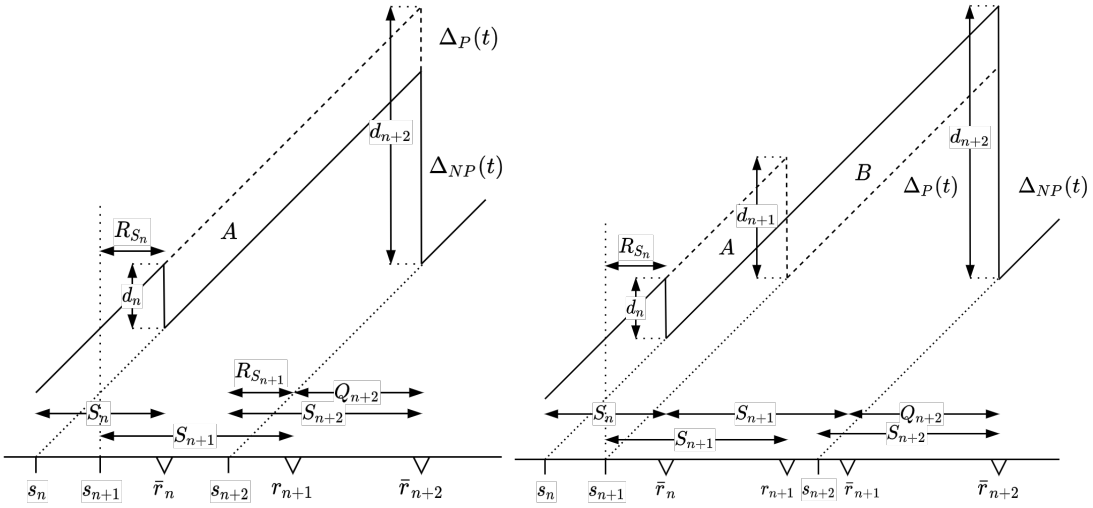

Figure 4.3: Age sample paths $\Delta_P(t)$ and $\Delta_{NP}(t)$ at left (at right) for the case, that the n^{th} packet is preempted (not preempted) by $(n+1)^{th}$ packet, at the same time, $(n+1)^{th}$ is also preempted by $(n+2)^{th}$ packet, which is delivered at $\bar{r}_{n+2}$, given that $R_{S_n} \leq S_{n+1}$.

However, calculating the exact expectation of preemption cost requires considering many cases that do not always have high probabilities for many service and arrival distributions. Since the preemption is problematic in terms of AoI for NBU-type service distributions, we will focus on these types of distributions. Therefore, in this work, we will calculate an approximate expectation of preemption cost for NBU-type service distributions.

Let us define the following probabilities to distinguish the events, which will be useful when calculating the expected cost of preemption:

$$\beta \coloneqq \mathrm{P}(R_{S_n} \leq S_{n+1} | \phi_{n+1} = 1) \tag{4.17a}$$

$$\gamma \coloneqq \mathrm{P}(X_{n+2} < R_{S_n} | \phi_{n+1} = 1, R_{S_n} \leq S_{n+1}) \tag{4.17b}$$

$$\sigma \coloneqq \mathrm{P}(R_{S_n} \leq X_{n+2} < S_{n+1} | \phi_{n+1} = 1, R_{S_n} \leq S_{n+1}) \tag{4.17c}$$

$$\rho \coloneqq \mathrm{P}(S_{n+1} \leq X_{n+2} < S_{n+1} + R_{S_n} | \phi_{n+1} = 1, R_{S_n} \leq S_{n+1}) \tag{4.17d}$$

Given (4.16) and the probabilities (4.17), the approximate expected values of $\mathbf{A_n}$ and

$\mathbf{B_n}$ are calculated as

$$\begin{aligned}\mathrm{E}[\mathbf{A}_n(d_n, d_{n+1}, \tau_{S_n})] \approx (1-\Phi_{n+1})d_n E[Q_{n+1}|\tau_{S_n}] \\ +\Phi_{n+1}\beta\sigma d_n(E[Q_{n+1}|\tau_{S_n}] + \mathrm{E}[Z]) \\ +\Phi_{n+1}\beta\gamma\mathrm{E}[\mathbf{A}_n(d_n, d_{n+2}, \tau_{S_n} + X_{n+2})],\end{aligned} \tag{4.18}$$

and

$$\begin{aligned}\mathrm{E}[\mathbf{B}_n(d_n, d_{n+1}, \tau_{S_n})] \approx (1-\Phi_{n+1})(d_{n+1}-d_n) E[R_{S_n}|\tau_{S_n}] \\ +\Phi_{n+1}\beta\rho(d_{n+1}-d_n)(E[R_{S_n}|\tau_{S_n}] + \mathrm{E}[Z]) \\ +\Phi_{n+1}\beta\gamma\mathrm{E}[\mathbf{B}_n(d_n, d_{n+2}, \tau_{S_n} + X_{n+2})],\end{aligned} \tag{4.19}$$

where Z is the time passed since the successive preemptions, whose number is assumed to be according to a geometric distribution with the preemption probability Φ, and $\mathrm{E}[Z] = \frac{\Phi}{1-\Phi}E[Q]$.

Based on the approximate expectations given in (4.18) and (4.19), we can define a preemption policy given as follows:

Definition 4.4.2. *Opportunistic Preemption policy (OP) is a preemption policy for LCFS queues, decides whether the new arrival preempts the ongoing service according to the criteria*

$$0 \lessgtr_{preempt}^{not\ preempt} \mathrm{E}[\mathbf{A}_n - \mathbf{B}_n | d_n, d_{n+1}, \tau_{S_n}], \quad n \in [1, N], \tag{4.20}$$

where d_n is the age difference of the ongoing service, d_{n+1} is the age difference of the prospective packet, and τ_{S_n} is the time passed since the service of n^{th} packet started. Ties are broken arbitrarily.

As you may realize, computing the expectation in (4.20) is challenging for many cases. In the following, we will examine OP policy under exponential and Erlang-2 service time distributions, where the expectations (4.18) and (4.19) can be further simplified.

In the case of i.i.d exponential service times together with Poisson arrivals, due to memoryless property of exponential distribution the expected residual time is equal to the expected service time, which is time independent and given as

$$\mathrm{E}[R_{S_n}(\tau_{S_n})] = \mathrm{E}[R_S] = \mathrm{E}[S_{n+1}] = \mathrm{E}[S] = \frac{1}{\mu}, \tag{4.21}$$

which yields

$$\mathrm{E}[Q_{n+1}] = \mathrm{E}[Q] = \mathrm{E}[R_S] - \mathrm{E}[S] = 0. \tag{4.22}$$

Hence, (4.20) becomes as follows

$$\begin{aligned} 0 &\lesseqgtr_{preempt}^{not\ preempt} \mathrm{E}[\mathbf{A}_n - \mathbf{B}_n | d_n, d_{n+1}] \\ &= \gamma\beta\Phi\mathrm{E}[\mathbf{A}_n - \mathbf{B}_n | d_n, d_{n+2}] - \rho\beta\Phi(d_{n+1} - d_n)E[R_{S_n}]. \end{aligned} \tag{4.23}$$

In (4.23), the second term of the right side of the equation is negative, if $d_{n+1} > d_n$. Additionally, since $\mathrm{E}[\mathbf{A}_n - \mathbf{B}_n | d_n, d_{n+1}]$ is a decreasing with respect to d_{n+1}, the first term is also negative, if $d_{n+1} > d_n$. Because $d_{n+1} > d_n$ for $n \in [1, \infty]$, we can define the OP policy for M/M/1 queue with a sufficient statistic, that

$$d_{n+1} \lesseqgtr_{preempt}^{not\ preempt} d_n, \quad n \in [1, N]. \tag{4.24}$$

Note that for M/M/1 queues, OP policy corresponds to regular preemption policy, that always preempts ongoing service upon a new arrival.

Similarly, in the case of i.i.d Erlang-2 service times together with Poisson arrivals, the expected residual time is time-independent, and computed as:

$$\mathrm{E}[R_{S_n}(\tau_{S_n})] = \mathrm{E}[R_S] = \frac{1}{2}\mathrm{E}[S_{n+1}] = \frac{1}{2}\mathrm{E}[S] = \frac{1}{\mu}, \tag{4.25}$$

which yields,

$$\mathrm{E}[Q_{n+1}] = \mathrm{E}[Q] = \mathrm{E}[S] - \mathrm{E}[R_S] = \frac{1}{\mu}. \tag{4.26}$$

When (4.25) and (4.26) are substituted in (4.20), the OP policy for Erlang-2 service times is computed as

$$\begin{aligned} &\frac{1-\Phi}{\mu}(d_{n+1} - 2d_n) + \beta\frac{\Phi}{(1-\Phi)\mu}[\rho d_{n+1} - (\rho + \sigma)d_n] \\ &+\beta\gamma\Phi\mathrm{E}[\mathbf{B}_n - \mathbf{A}_n | d_n, d_{n+2}] \lesseqgtr_{preempt}^{not\ preempt} 0, \quad n \in [1, N]. \end{aligned} \tag{4.27}$$

Although, γ, σ, and ρ in (4.27) don't depend on τ_{S_n}, they depend on the arrival process, even if we assume Poisson arrival process. Therefore, we propose a preemption policy for Erlang-2 service time distribution and Poisson arrivals that depends on the arrival rate λ, defined as follow:

$$d_{n+1} \lesseqgtr_{preempt}^{not\ preempt} \theta(\lambda)d_n, \quad n \in [1, N], \tag{4.28}$$

where $\theta(\lambda) > 1$ for $\lambda \in \mathbb{R}^+$. In section 4.5, we provide results for this preemption policy, when $\theta(\lambda) = 2$, which ensures that the first term in (4.27) is nonnegative.

Now, we can generalize OP policy for multiple sources with the following Lemma.

Lemma 3. *Given K sources, each receives Poisson arrivals with rate λ_i, Opportunistic Preemption policy is a causal work-conserving preemption policy improves the average system age $\bar{\Delta}$, which takes preemption decision based on the following formula*

$$0 \lesseqgtr_{preempt}^{not\ preempt} \mathrm{E}[\mathbf{A} - \mathbf{B}|d, d', \tau_S], \tag{4.29}$$

where d is the age difference of the ongoing service, d' is the age difference of the prospective packet, the random variable S denotes the service time, and τ_S is time passed since the ongoing service started. Ties are broken arbitrarily.

4.4.2 Preemptive MAD Scheduling

Since we have come up with a solution to preemption problem and nonpreemptive scheduling problem, we can combine these solutions to propose a preemptive scheduling policy.

Definition 4.4.3. *Maximum Age Difference - Opportunistic Preemption (MAD-OP) is a preemptive scheduling algorithm, which decides the next flow to be served among K flows, according to formula*

$$i^*(t) = \arg\max_i d_i(t), \quad i \in \{1, \ldots, K\}, \tag{4.30}$$

when the system is idle, and it preempts the ongoing service if new arrival satisfies

$$\mathrm{E}[\mathbf{B}_n|d_n, d_{n+1}, \tau_{S_n}] \geq \mathrm{E}[\mathbf{A}_n|d_n, d_{n+1}, \tau_{S_n}]. \tag{4.31}$$

where d_n is the age difference of the ongoing service, d_{n+1} is the age difference of the prospective packet, the random variable S denotes the service time, and τ_{S_n} is time passed since the ongoing service started. Ties are broken arbitrarily.

In the next section, we will provide the simulation results of MAD policy comparing with the MAF policy under preemptive and nonpreemptive service disciplines.

4.5 Simulation Results

In this section, we provide the simulation results for the Maximum Age Difference (MAD) and Opportunistic Preemption (OP) policies under preemptive and nonpreemptive service disciplines, together with exponential and Erlang-2 distributed service times. The performance of the MAD policy is investigated by comparing it with *Maximum Age Difference* (MAF) policy. The MAF is a scheduling algorithm proposed in [25]. MAF policy schedules the flow which has the highest instantaneous age. In [25, 59], the optimality of the MAF algorithm is shown for synchronous flows under preemptive service discipline, and for *generate-at-will* model, that source generates a new packet just before sending, together with an optimal sampler. In our simulation, we compared the performances of MAF and MAD algorithms under a more general setup, that considers asynchronous flows, which have independent Poisson arrivals and i.i.d service times, and considers both Erlang and exponential distributed services times.

The simulation results are obtained by using the Python-Network-Simulator[1]. For each Monte-Carlo simulation, 30000 packet arrivals for each flow are calculated. The simulation setting used for this study composed of Intel Core i7-4650U (4 core, up to 3.30 GHz), 8 GB RAM.

In Figure 4.4, the Opportunistic Preemption policy is compared with the regular preemptive policy, which always preempts when an arrival hits during service, and the nonpreemptive policy, which never preempts. As discussed before, the preemption policy affects the AoI performance differently under various service time distributions. In Figure 4.4, these preemption policies are investigated under exponential and Erlang-2 distributed service times, Poisson packet arrivals and unlimited buffer. In the figure, OP, P and NP represent Opportunistic Preemption policy, Preemptive policy and Nonpreemptive policy, respectively. The dashed and straight lines show the results for exponential and Erlang-2 service time distributions. The main focus of this figure is that even a sub-optimal OP policy (defined in (4.28)) used with $\theta(\lambda) = 2$ for

[1] Python-Network-Simulator is a Python-based modular continuous time event simulator, specifically designed for Age-of-Information research, and able to simulate various sampling and scheduling scenarios, which may contain different buffer management, queue disciplines, scheduling algorithms, multi/single-source network topologies, and service/arrival processes. Because of the modular design, it is easy to add new configurations and settings with minimum effort. Available at github.com/hbbeytur/Python-Network-Simulator

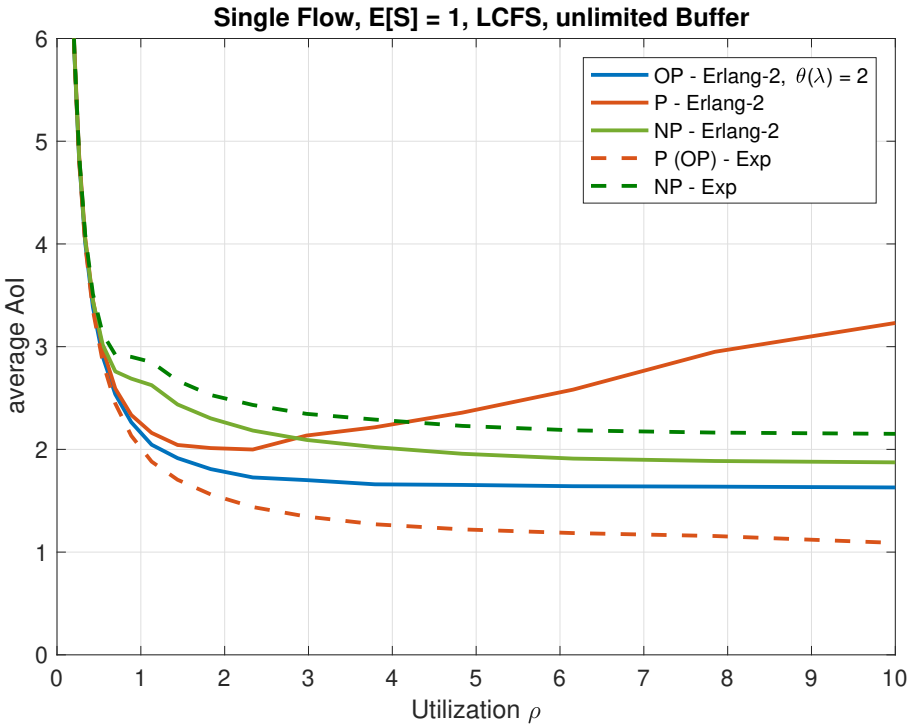

Figure 4.4: The average age versus utilization graphs for Opportunistic Preemption policy (blue solid line), regular preemption policy (orange lines) which always preempts and nonpreemptive policy (green lines), which never preempts under exponential and Erlang-2 distributed service times. The arrivals are assumed to be Poisson.

Erlang-2 distribution outperforms both P and NP policies, and the average age does not grow even at high throughput when OP ($\theta(\lambda) = 2$) policy is used. For exponential service time distribution, the OP policy corresponds to the regular preemptive policy (P policy).

In Figure 4.5, the MAD policy is compared with the Maximum Age First (MAF) policy under the nonpreemptive service discipline. The results are obtained from a simulation for a network with four independent sources receiving Poisson arrivals with equal rates $\lambda_i,\ i \in [1, 4]$ and having buffers with a buffer size of 1-packet. In the simulation, exponential and Erlang-2 distributed service times with a mean of $\mathrm{E}[S] = 1$ are used. The x-axis represents the load per-source $\rho_i = \lambda_i/\mu$. The results verify the Theorem 6, which states the optimality of MAD policy among all nonpreemptive, work-conserving and causal policies. Though, as seen in the figure, MAF policy

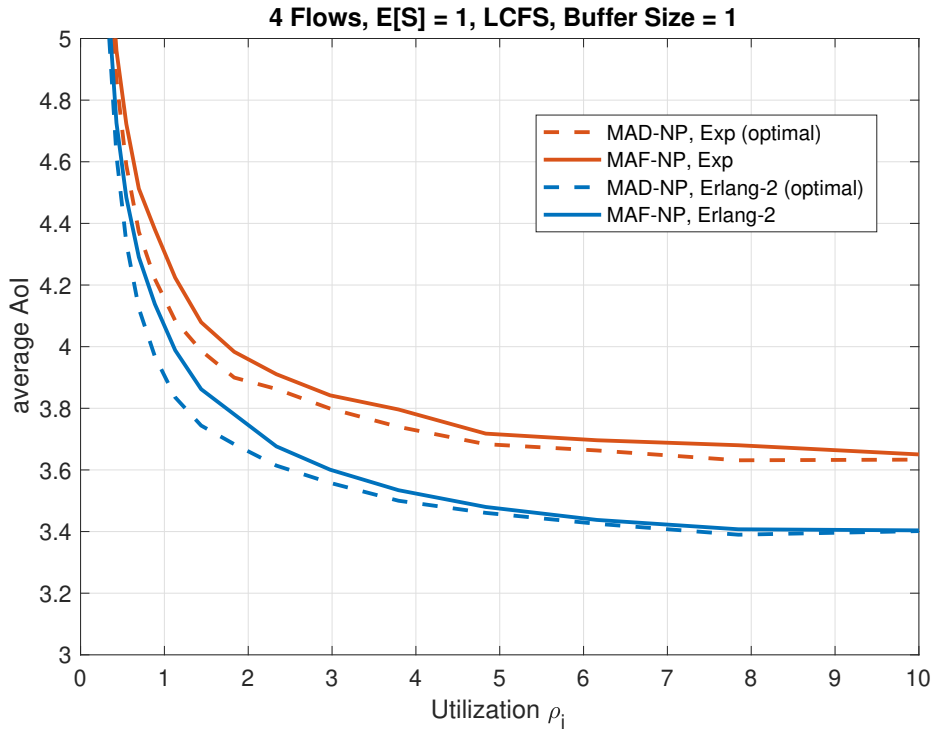

Figure 4.5: Average age versus utilization results for MAD-NP (dashed lines) and MAF-NP (solid lines) policies under exponential (orange) and Erlang-2 (blue) distributed service times with Poisson arrivals.

performs very close to the optimal for the cases considered in the simulation.

In Figure 4.6, the average age performance of MAD, MAD-OP and MAF policies under preemptive service discipline is illustrated. Similar to previous figures, the policies are compared with each other when service time distribution is exponential and Erlang-2 (NBU), and packets arrive according to independent Poisson processes. The dashed lines represent the average age results of MAD-P policy. Since MAD-OP policy corresponds to MAD-P policy under exponential service time distribution and Poisson arrivals, a single result represents both policies. As seen in the Figure 4.6, MAD-P and MAF-P policies have similar results with a small difference. However, under Erlang-2 distributed service times, MAD-OP policy's result distinguishes from others substantially, due to the novel preemption policy. Although the result (blue line) obtained by a sub-optimal MAD-OP policy, which uses a constant OP parameter $\theta(\lambda) = 2$, it outperforms MAD-P and MAF-P, especially at high load. Similar to OP

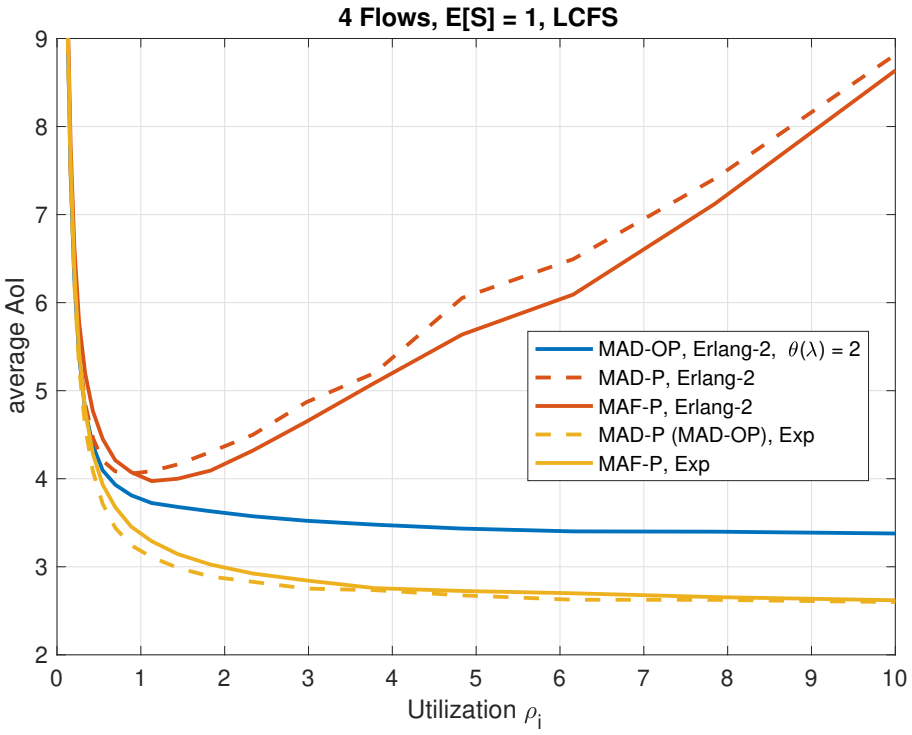

Figure 4.6: Average age versus utilization results for MAD-OP (blue), MAD-P (dashed lines) and MAF-P (solid lines) policies under exponential (yellow) and Erlang-2 (orange) distributed service times with Poisson arrivals.

policy in Figure 4.4, MAD-OP policy avoids the wrong preemption decisions which lead to an increase in average at high load when service time distribution is NBU. The average age increases when MAD-P and MAF-P are employed under Erlang-2 service distribution. Interestingly, before the average age increases with increasing throughput, MAD-P yields a lower average age than MAF-P. However, after that, both policies show an increasing trend, and MAF-P outperforms MAD-P.

In the following simulation, the main focus is the AoI behavior as the number of flows increases. In other words, from an AoI perspective, should users be allocated individual resources (bandwidth) or should they share the pooled resources of the server? In order to answer this question, we simulated the case of increasing the number of flows while the total traffic is constant.

According to simulation results shown in Figure 4.7, due to flow diversity, although the same resources are used (bandwidth, etc.), MAD policy yields lower average age

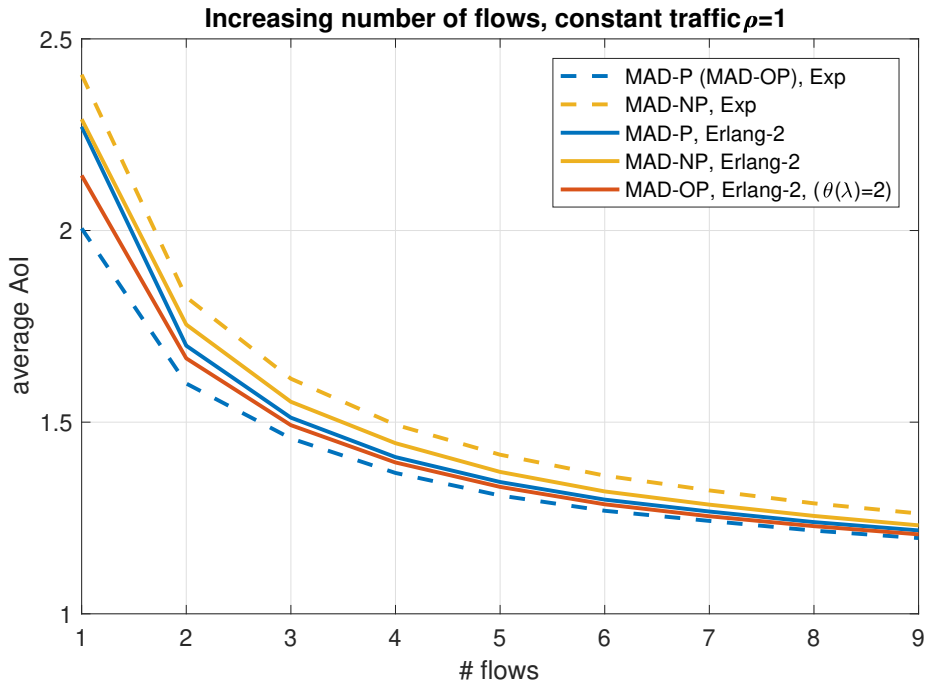

Figure 4.7: Average age versus number of flows results under MAD policy, constant traffic $\rho = 1$

values for all preemption policies NP, P and OP. As seen in Figure Figure 4.7, MAD-OP policy results in the lowest average age among other policies for both exponential and Erlang-2 distributed service.

CHAPTER 5

REINFORCEMENT LEARNING BASED SCHEDULING

In this chapter, we employed the Reinforcement Learning algorithms to solve the multi-user scheduling problem, which is formulated as a multi-armed bandit problem. This problem formulation is inspired from [46], which used a variation of SARSA algorithm for scheduling packet transmissions on a link using Hybrid ARQ. In this work, we used Deep Q-Network (DQN) and Policy Gradients methods to solve the scheduling problem.

In section 5.1, the mathematical background of Reinforcement Learning together with the details related to Deep Q-Network and Policy Gradients is given. Then, in section 5.2, the discrete time network model is introduced. Finally, in section 5.3, the average age performance results of scheduling method using DQN and Policy Gradients are compared with each other and deterministic policies MAD and MAF, discussed in chapter 4.

5.1 Mathematical Background on Reinforcement Learning

Reinforcement Learning is a class of machine learning algorithms which learns to control the action in an environment formulated as a Markov Decision Process defined by 4-tuple $(\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S}$ is the set of states called *state space*, $\mathcal{A}$ is the set of actions called *action space*, P is the transition probability and R is the reward function.

A general interaction scheme between agent and environment is given in Figure 5.1. At a time step t, the agent observes the environment, while it is in state $s_t \in \mathcal{S}$.

By using the previous experience, the agent reacts to the current state with action $a_t \in \mathcal{A}$ and as a result of that action agent reaches to the next state $s_{t+1} \in \mathcal{S}$ and produces a reward r_{t+1}. The state transition occurs according to the dynamics of the environment, and the reward signal is the main objective of the agent. Briefly, a reinforcement learning agent learns to interact with the environment to get higher rewards continuously.

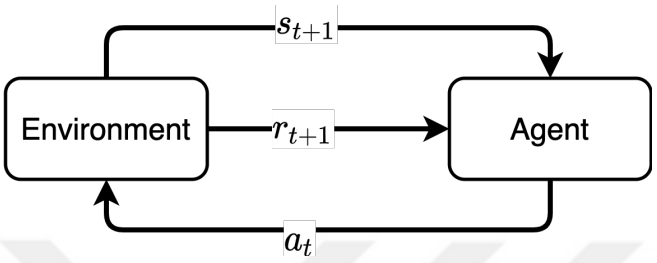

Figure 5.1: Agent-Environment Interaction in Reinforcement Learning

The state transitions and the immediate rewards in an MDP problem are defined as

$$P(s' \mid s, a) = \Pr\left\{s_{t+1} = s' \mid s_t = s, a_t = a\right\}, \tag{5.1a}$$

$$r_{t+1} = \mathrm{R}(s_t, a_t, s_{t+1}), \tag{5.1b}$$

where the subscript t and $t+1$ denote the current and next time step. The next state in an MDP is conditioned on the current state-action pair. This is called *Markov Property*, that the next state only depends on the current state and action, not the previous ones. Similarly, the immediate reward r_{t+1} is supplied by the reward function R depends only on the current state-action pair.

At the beginning of the learning, the agent has no a priori information on how to act. Therefore, the agent explores the state-action pairs returning high reward. However, the action taken in the current state effects not only the immediate reward but also the future rewards through the next states. To overcome this challenge the notion of *discounted reward* is employed. The discounted reward at time step t is defined as the discounted sum of the future reward until the end of the episode at time step T,

i.e.,

$$G_t = r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t} r_T = \sum_{k=0}^{T-t} \gamma^k r_{t+k} \tag{5.2}$$

where $\gamma \in [0, 1]$ is the *discount factor*, that trades between the importance of the short-term and the long-term rewards.

To determine the optimal policy at a state $s \in \mathcal{S}$, we use *action-value function* (or *Q-function*) which is defined as the expected discounted reward at a given state s and action a, taken according to a policy $\pi \in \Pi$, such that

$$Q^\pi(s, a) = \mathrm{E}[G_t | s_t = s, a_t = a, \pi], \tag{5.3}$$

where the optimal policy $\pi^* \in \Pi$ yields that

$$Q^*(s, a) = \max_{\pi \in \Pi} Q^\pi(s, a), \quad \forall s \in \mathcal{S} \tag{5.4}$$

However, without knowing the system dynamics and reward function, we cannot calculate the optimal Q-values. Therefore, we use the Bellman Equation to put the Q-function in a form that it can be calculated recursively, and it is given as follows.

$$Q^\pi(s, a) = \sum_{s' \in \mathcal{S}} P(s' \mid s, a) \Big[R(s, a, s') + \gamma \max_{a' \in \mathcal{A}} Q^\pi(s', a') \Big] \tag{5.5}$$

There are several Reinforcement Learning methods that solve (5.5). One of the well-known methods is the Q-Learning [60] that stores the Q-values on a table of size $\mathcal{S}x\mathcal{S}$, and updates the table at every iteration according to the resulting immediate reward. It is shown in [60] that Q-Learning method converges to the optimal policy. Although the tabular methods perform very well with the problems with low-dimensional state and action spaces, they are not useful for the problems with high dimensional or continuous state and action spaces. For this kind of problems, there are RL algorithms that use function approximation to fit the Q-function. One way of function approximation is to use neural networks, and the methods using neural networks to approximate the action-value function form a subclass of algorithms called Deep Reinforcement Learning.

In the following subsection, we will discuss two common Deep Reinforcement Learning algorithms, called Deep Q-Network and Policy Gradients.

5.1.1 Deep Q-Network

Deep Q-Network (DQN), introduced in [61], is a value-based algorithm that constructs the policy based on the value function, and the value function is formulated by a neural network.

In DQN, the learning agent's experiences are stored as 4-tuples $< s, a, r, s' >$, which are current state, current action, immediate reward and the next state of the transition. These experiences the agent collected are used to train the value network by using gradient descent algorithms like in the supervised learning.

The algorithm starts with the initial parameters θ_0 of action-value network that corresponds to action-value function $Q(s, a; \theta_0)$. At time step k, the target action-value is calculated by Bellman Equation, i.e.,

$$Q_k^* = r_k + \max_{a'} Q(s', a'; \theta_k), \tag{5.6}$$

where θ_k is the network parameters at time step k. Then, we define a loss function, which gives square error between the result of the Bellman equation and the current Q-value.

$$L(\theta_k) = \frac{1}{2}[Q_k^* - Q(s_k, a_k; \theta_k)]^2. \tag{5.7}$$

Given the loss function (5.7), the parameters are updated by using gradient descent, such that;

$$\theta_{k+1} = \theta_k - \alpha \nabla_{\theta_k} L(\theta_k), \tag{5.8a}$$

$$\theta_{k+1} = \theta_k + \alpha (Q_k^* - Q(s_k, a_k; \theta_k)) \nabla_{\theta_k} Q(s_k, a_k; \theta_k), \tag{5.8b}$$

where α is the scalar learning rate.

The content covered in this subsection gives the essentials of the DQN algorithm. There are many variants of DQN improving the performance, but there are out of the scope of this subsection.

5.1.2 Policy Gradients

While the goal of Q learning is to approximate the Q function and use it to infer the optimal policy π^*, Policy Gradients seeks to optimize directly in the policy space. In Policy Gradients, a neural network is used to model the probability of action given the state directly. At each time step, the agent interacts with the environment, obtains data points $< s, a, r, s' >$, and by using these data points updates the network parameters to sample the actions more likely to be *good* in the future. This is done until the optimal policy π^* is reached.

The Policy Gradients method tries to maximize the expected discounted reward $\mathrm{E}[G_t]$ under a parametrized policy $\pi_\theta(s, a)$. The maximization problem is given as follows.

$$\underset{\theta}{\text{maximize}}\ \mathrm{E}[f(x)] = \mathrm{E}_{\pi_\theta}[G_t] \tag{5.9}$$

Although various Policy Gradients algorithms optimize different functions instead of the expected discounted reward, in this subsection, we will explain the algorithm by using the expected discounted reward. Similar to DQN, the parameters of the policy network is updated by performing gradient ascent on θ by using the gradient of the objective function.

$$\theta \leftarrow \theta + \alpha \nabla_\theta \mathrm{E}_{\pi_\theta}[G_t] \tag{5.10}$$

where α is the scalar learning rate.

The derivative of objective is computed as given below.

$$\nabla_\theta \mathrm{E}[f(\theta)] = \nabla_\theta \int p_\theta(x) f(x) dx \tag{5.11a}$$

$$= \int f(x) p_\theta(x) \nabla_\theta \log p_\theta(x) dx \tag{5.11b}$$

$$= \mathrm{E}[f(x) \nabla_\theta \log p_\theta(x)], \tag{5.11c}$$

where p_θ is the density function of x.

The policy can be updated step by step by estimating the gradient. An important issue encountered in Policy Gradients is the high variance in the gradient estimates. To reduce the high variance in the gradient estimate, the objective function is subtracted by a *baseline function*, which can be formulated an additional neural network. The

algorithms use both policy and value networks are called actor-critic algorithms, and they are a mixture of the value-based and policy-based algorithms.

In this study, we used these two algorithms to train a scheduling agent on a slotted-time multi-user system model, explained in the next section.

5.2 System Model

We consider a network model with a single server receiving updates from multiple sources, similar to the model in Figure 4.1 in Chapter 4. In practice, these sources can correspond to various sensor nodes, measuring different physical quantities. Each source-destination pair has its own age process, and the server needs to schedule among these update flows to achieve the minimum average age of information. We assume that packets are generated at sources according to discrete time stochastic processes and put into queues. Additionally, we assume that there is no delay between sources and server.

We consider that the time is slotted, and t_n denotes the n^{th} time step. Furthermore, the packets are assumed to be generated according to i.i.d Bernoulli processes that each source receives a new packet arrival with probability $p_{\lambda,k}$ at each time slot. Also, the server can serve at most one flow at any time step. Therefore, at each time step, the scheduling decision is either to serve one of the flows or to stay idle. We assume that when the server schedules a packet, the transmission is successful with a probability p_μ. If the server fails to transmit the packet, it is placed back into its queues.

At any t_n, the timestamp of the freshest packet received by the k^{th} destination is denoted by U_n^k. The age at time step t_n for k^{th} flow is

$$\Delta^k(t_n) = t_n - U_n^k, \tag{5.12}$$

and the expected average age of information belonging to the k^{th} flow is

$$\Delta^k = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N} \Delta^k(t_n). \tag{5.13}$$

Similarly, the system age which is the average age over all K flows is calculated as

$$\Delta = \frac{1}{K} \sum_{k=1}^{K} \Delta^k. \tag{5.14}$$

Instantenous Age, FCFS Policy

Sources																
	a	1	2	3	4	5	5	6	7	8	9	8	5	6	6	7	8
	b	1	2	2	3	4	5	2	3	4	5	6	2	3	4	5	6
	c	1	1	2	3	2	3	4	5	6	2	3	4	5	6	1	2
	d	1	2	3	3	4	5	6	2	3	4	5	6	7	8	9	3
	timesteps	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

(a)

Instantenous Age, LCFS Policy

Sources																
	a	1	2	3	4	5	3	4	5	6	7	3	4	5	2	3	4
	b	1	2	2	3	4	5	2	3	4	5	6	2	3	4	5	6
	c	1	1	2	3	2	3	4	5	6	2	3	4	5	6	1	2
	d	1	2	3	3	4	5	6	2	3	4	5	6	7	8	9	3
	timesteps	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

(b)

Figure 5.2: Changes in instantaneous age of multiple flows in a slotted update system with (a) FCFS, (b) LCFS service policy. Scheduling decisions are shown with red and packet generation instants are shown with blue.

In Figure 5.2, the changes in instantaneous age for an arbitrary sequence of schedule decisions are given for a sample network with four flows. Packet generation instants are indicated with blue color. The scheduling decisions are shown with red color. Figure 5.2a indicates the AoI process when the server employs a FCFS policy, and Figure 5.2b indicates the result of a LCFS policy. The difference between the LCFS and FCFS queue disciplines can be observed by viewing the age process of Source a. As sources c and d have empty queues, scheduling these sources at time steps 13 and 9 does not yield a transmission and consequently, the ages of these sources increase. After a successful transmission, the age drops to the time difference between transmission and generation time of the transmitted packet.

Let $q_t^i \in (0, .., q_{max})$ denote the number of packets waiting in the queue of the k^{th} flow, where q_{max} is the buffer size. In our framework, the server is the learning agent

and we assume that it can track the individual instantaneous age values of flows (as it knows about the most recent successful transmission event and the timestamp of the packets sent) and the number of packets waiting in the queues of flows. These two vectors form the state of the learning agent $s_t \stackrel{\Delta}{=} (\Delta_t^1, q_t^1, ...\Delta_t^K, q_t^K)$, where K is the number of flows. At each state, the server decides which flow to serve. The action set includes the union of the set of flows, and the *idle* action, which corresponds to serving no flow. Correspondingly, the cardinality of the action space equals to one more than the number of flows and the cardinality of the observation space is equal to twice the number of flows. The reward received after the action is calculated as the negative sum of the instantaneous ages of all flows. The agent is trained on the discounted rewards. The reward and state-space formulation bear similarity to that in [46], with the exception that the said study is concerned with a link with ARQ/HARQ.

The scheduler agent is trained in this environment by DQN and Policy Gradients algorithms. In our setting, the neural network in the Policy Gradients has an additional hidden layer with 10 nodes, learning rate of 0.02 and discount factor of 0.99. For the case of DQN, there are two neural networks for prediction and target parts with 2 hidden layers with 10 nodes, the learning rate is 0.01.

5.3 Performance Comparison between DQN and Policy Gradient

In this part, we will give the training results of DQN and Policy Gradients with LCFS and FCFS service policies. In our network, there are four flows, the arrival probabilities are 0.1, 0.2 in separate simulations. The transmission probability is equal to 1, which means that the transmission channel is assumed to be error-prone.

In Figure 5.3, the average age values obtained throughout the training process by using Deep Q-Network (DQN) and Policy Gradients (PG) algorithms can be seen. The results of MAD and MAF algorithms are put in the figure as references. Although the resulting average ages under MAD or MAF algorithms fluctuate in episodes due to the changing samples of arrival and service probabilities, they give us an idea of how those algorithms perform. As discussed in Chapter 4, MAD algorithm outperforms

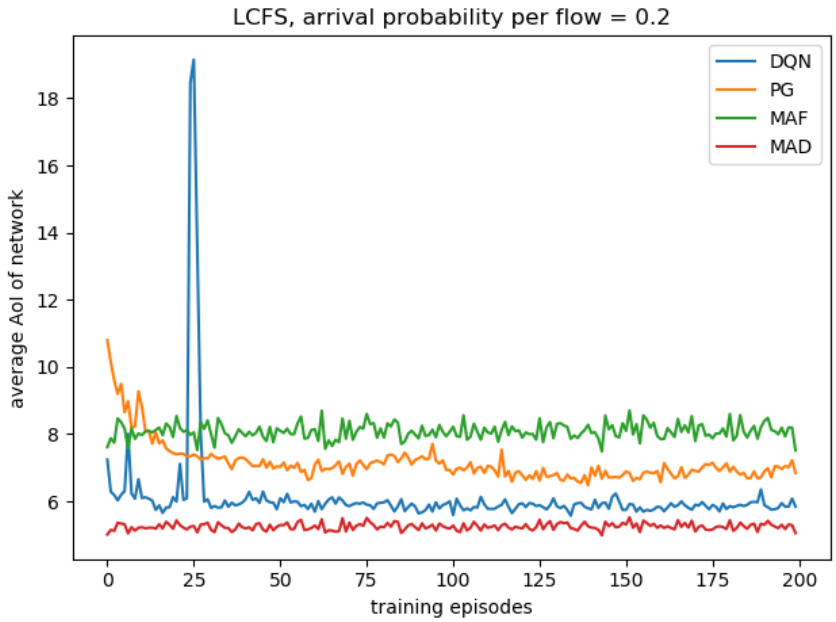

Figure 5.3: Average age of information versus training episodes, LCFS queue, arrival probability = 0.2

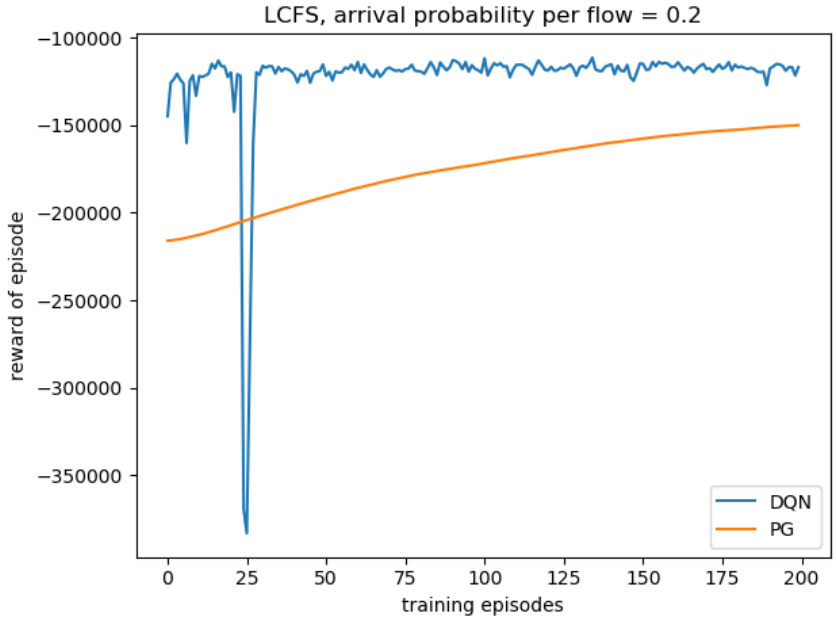

Figure 5.4: Episode rewards, LCFS queue, arrival probability = 0.2

MAF algorithm under many circumstances. In these tests on discrete time network model, this result does not change.

As you can see, DQN results in average AoI values very close to the MAD algorithm. Although the PG algorithm gets better in time, it cannot reach to DQN. It is interesting that, although the DQN, PG and MAF algorithms use the same instantaneous age information, both, DQN and PG outperform MAF. This result shows the unexpected performance of Reinforcement Learning. However, DQN and PG cannot outperform MAD. We argue that the additional information about the timestamp of the latest received packet MAD uses is quite useful for age minimization.

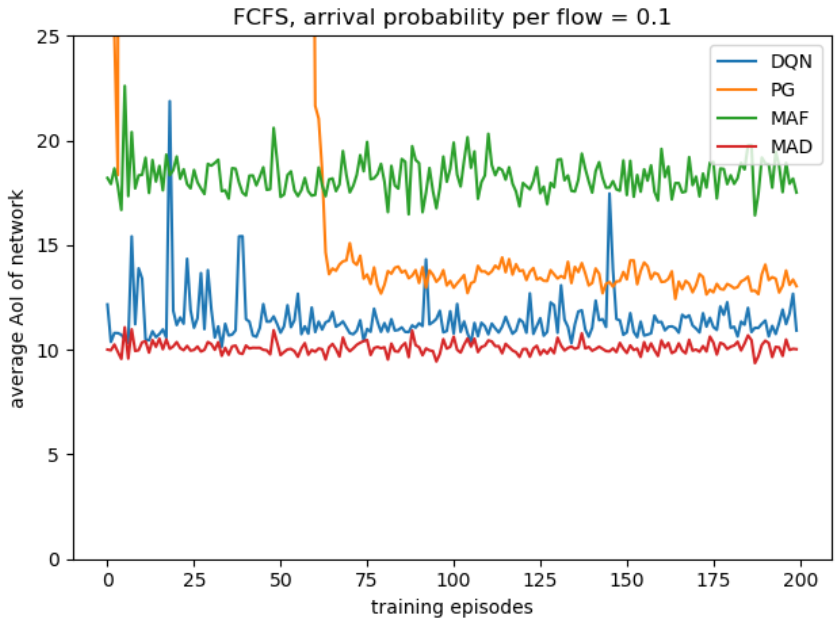

Figure 5.5: Average age of information versus training episodes, FCFS queue, arrival probability = 0.1

You can see the rewards of the DQN and PG in the training process in Figure 5.4. As you can see, although the DQN is computationally expensive, it takes less time to convergence for DQN.

In Figure 5.5 and Figure 5.6, the training results for the FCFS network can be seen. It is known that FCFS systems perform worse than the LCFS system in terms of AoI. This result is confirmed in our simulations.

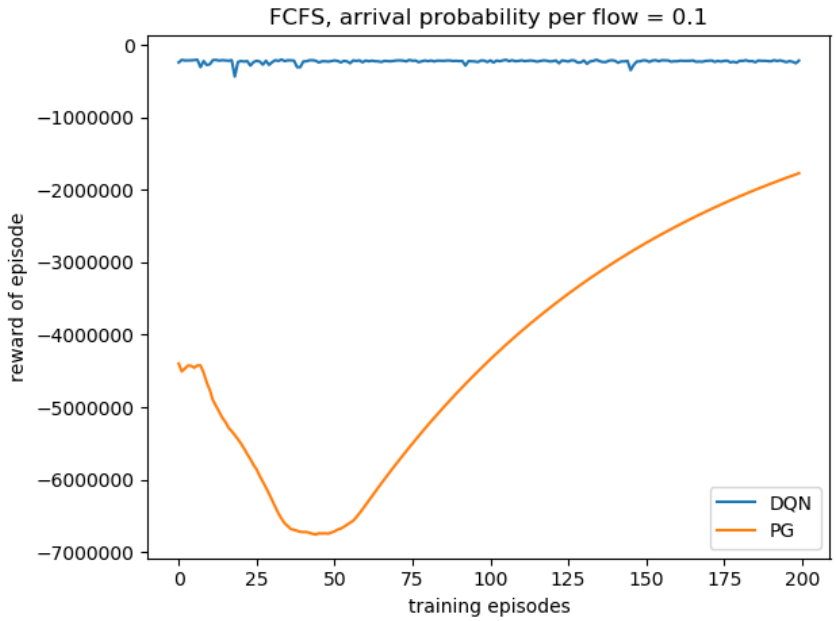

Figure 5.6: Episode rewards, FCFS queue, arrival probability = 0.1

You can see that the ordering of the algorithms according to average AoI is similar to the LCFS case. Note that the arrival probability is taken to be 0.1, because the PG algorithm failed to converge when arrival probability is higher. This also shows us the vulnerability of Policy Gradients method.

CHAPTER 6

CONCLUSION

In this thesis, we investigated the age-aware scheduling problem on various network models by using different techniques.

Firstly, in chapter 3, we used SHS technique to study a multi-source network model, where each source has its own 1-packet buffers. By computing the closed-form average age expressions for network models, we showed that the 1-packet buffer each source has can improve the achievable age region for LCFS queues. Additionally, we also proposed an SHS framework that can be used to model the network with hypo-exponential service distribution. We used this framework to compute the closed-form expression of average age for the models with Erlang service distribution.

Secondly, in chapter 4, we proposed a scheduling policy called Maximum Age Difference (MAD). We showed that MAD policy is optimal under nonpreemptive service discipline for general service and arrival distributions. Additionally, by investigating the preemption in LCFS, we proposed a novel preemption policy called Opportunistic Preemption, which avoids bad preemption decisions in terms of average AoI. By using Opportunistic Preemption policy, we came up with a new preemptive scheduling policy called MAD-OP, which improves the average age performance, especially for NBU-type service distributions.

Lastly, in chapter 5, we used two of Reinforcement Learning algorithms, namely DQN and Policy Gradients methods, to solve the age-minimizing scheduling problem. We see that DQN performs quite well even though it does not have any prior information on network statistics. The results are very promising and show the potential of Reinforcement Learning in solving scheduling problems.

REFERENCES

[1] H. B. Beytur, S. Baghaee, and E. Uysal, "Measuring Age of Information on Real-Life Connections," in *2019 27th Signal Processing and Communications Applications Conference (SIU)*, (Sivas, Turkey), pp. 1–4, IEEE, Apr. 2019.

[2] H. B. Beytur, S. Baghaee, and E. Uysal, "Towards AoI-aware Smart IoT Systems," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, (Big Island, HI, USA), pp. 353–357, IEEE, Feb. 2020.

[3] H. B. Beytur and E. Uysal, "Age Minimization of Multiple Flows using Reinforcement Learning," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, pp. 339–343, Feb. 2019. tex.number: ISSN: 2325-2626.

[4] H. B. Beytur and E. Uysal-Bıyıkoglu, "Minimizing age of information on multi-flow networks," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, May 2018. tex.number: ISSN:.

[5] H. B. Beytur and E. Uysal-Biyikoglu, "Minimizing Age of Information for Multiple Flows," in *2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 1–5, June 2018. tex.number: ISSN:.

[6] R. G. Gallager, *Stochastic Processes: Theory for Applications*. Cambridge University Press, 2014.

[7] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pp. 350–358, June 2011. ISSN: 2155-5494.

[8] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?," in *2012 Proceedings IEEE INFOCOM*, pp. 2731–2735, Mar. 2012.

[9] S. K. Kaul, R. D. Yates, and M. Gruteser, "Status updates through queues," in *2012 46th Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6, Mar. 2012.

[10] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "Age and value of information: Non-linear age case," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 326–330, June 2017.

[11] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff, "Update or Wait: How to Keep Your Data Fresh," *IEEE Transactions on Information Theory*, vol. 63, pp. 7492–7508, Nov. 2017.

[12] M. Klügel, M. H. Mamduhi, S. Hirche, and W. Kellerer, "AoI-Penalty Minimization for Networked Control Systems with Packet Loss," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 189–196, Apr. 2019.

[13] Y. Sun and B. Cyr, "Sampling for data freshness optimization: Non-linear age functions," *Journal of Communications and Networks*, vol. 21, pp. 204–219, June 2019. Conference Name: Journal of Communications and Networks.

[14] E. Najm and R. Nasser, "Age of information: The gamma awakening," in *2016 IEEE International Symposium on Information Theory (ISIT)*, pp. 2574–2578, July 2016.

[15] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, "Age of information with a packet deadline," in *2016 IEEE International Symposium on Information Theory (ISIT)*, (Barcelona, Spain), pp. 2564–2568, IEEE, July 2016.

[16] M. Costa, M. Codreanu, and A. Ephremides, "Age of information with packet management," in *2014 IEEE International Symposium on Information Theory*, pp. 1583–1587, June 2014. ISSN: 2157-8117.

[17] Y. Inoue, H. Masuyama, T. Takine, and T. Tanaka, "A General Formula for the Stationary Distribution of the Age of Information and Its Application to Single-Server Queues," *arXiv:1804.06139 [cs, math]*, Apr. 2018.

[18] R. D. Yates, "Lazy is timely: Status updates by an energy harvesting source," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 3008–3012, June 2015.

[19] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Age-optimal information updates in multihop networks," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 576–580, June 2017. ISSN: 2157-8117.

[20] Q. He, D. Yuan, and A. Ephremides, "Optimal Link Scheduling for Age Minimization in Wireless Systems," *IEEE Transactions on Information Theory*, vol. 64, pp. 5381–5394, July 2018. Conference Name: IEEE Transactions on Information Theory.

[21] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Minimizing the Age of Information in broadcast wireless networks," in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 844–851, Sept. 2016.

[22] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Scheduling Policies for Minimizing Age of Information in Broadcast Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 26, pp. 2637–2650, Dec. 2018.

[23] I. Kadota, A. Sinha, and E. Modiano, "Scheduling Algorithms for Optimizing Age of Information in Wireless Networks With Throughput Constraints," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2019.

[24] Y.-P. Hsu, E. Modiano, and L. Duan, "Scheduling Algorithms for Minimizing Age of Information in Wireless Broadcast Networks with Random Arrivals," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2019. Conference Name: IEEE Transactions on Mobile Computing.

[25] Y. Sun, E. Uysal-Biyikoglu, and S. Kompella, "Age-Optimal Updates of Multiple Information Flows," *arXiv:1801.02394 [cs, math]*, Mar. 2018.

[26] A. M. Bedewy, Y. Sun, S. Kompella, and N. B. Shroff, "Optimal Sampling and Scheduling for Timely Status Updates in Multi-source Networks," *arXiv:2001.09863 [cs, math]*, Jan. 2020. arXiv: 2001.09863.

[27] J. Zhong, W. Zhang, R. D. Yates, A. Garnaev, and Y. Zhang, "Age-aware Scheduling for Asynchronous Arriving Jobs in Edge Applications," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, (Paris, France), pp. 674–679, IEEE, Apr. 2019.

[28] R. Talak, S. Karaman, and E. Modiano, "Optimizing age of information in wireless networks with perfect channel state information," in *2018 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 1–8, May 2018.

[29] R. Talak, S. Karaman, and E. Modiano, "Optimizing Information Freshness in Wireless Networks under General Interference Constraints," in *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, (Los Angeles CA USA), pp. 61–70, ACM, June 2018.

[30] R. Talak, S. Karaman, and E. Modiano, "Distributed Scheduling Algorithms for Optimizing Information Freshness in Wireless Networks," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, June 2018. ISSN: 1948-3252.

[31] R. Talak, I. Kadota, S. Karaman, and E. Modiano, "Scheduling Policies for Age Minimization in Wireless Networks with Unknown Channel State," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2564–2568, June 2018. ISSN: 2157-8117.

[32] R. D. Yates and S. K. Kaul, "The Age of Information: Real-Time Status Updating by Multiple Sources," *arXiv:1608.08622 [cs, math]*, Aug. 2016.

[33] R. D. Yates, J. Zhong, and W. Zhang, "Updates with Multiple Service Classes," in *2019 IEEE International Symposium on Information Theory (ISIT)*, (Paris, France), pp. 1017–1021, IEEE, July 2019.

[34] R. D. Yates, "Status Updates through Networks of Parallel Servers," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2281–2285, June 2018. ISSN: 2157-8117.

[35] R. D. Yates and S. K. Kaul, "The Age of Information: Real-Time Status Updat-

ing by Multiple Sources," *IEEE Transactions on Information Theory*, vol. 65, pp. 1807–1827, Mar. 2019.

[36] R. D. Yates, "The Age of Information in Networks: Moments, Distributions, and Sampling," *arXiv:1806.03487 [cs, math]*, Oct. 2019.

[37] R. D. Yates, "Age of Information in a Network of Preemptive Servers," *arXiv:1803.07993 [cs, math]*, Mar. 2018.

[38] A. Maatouk, M. Assaad, and A. Ephremides, "Minimizing The Age of Information in a CSMA Environment," *arXiv:1901.00481 [cs, math]*, Jan. 2019. arXiv: 1901.00481.

[39] A. Maatouk, M. Assaad, and A. Ephremides, "Age of Information With Prioritized Streams: When to Buffer Preempted Packets?," *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 325–329, July 2019.

[40] S. K. Kaul and R. D. Yates, "Age of Information: Updates with Priority," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2644–2648, June 2018. ISSN: 2157-8117.

[41] A. Javani, M. Zorgui, and Z. Wang, "Age of Information in Multiple Sensing," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Dec. 2019. ISSN: 2576-6813.

[42] J. Doncel and M. Assaad, "Age of Information in a Decentralized Network of Parallel Queues with Routing and Packets Losses," *arXiv:2002.01696 [cs, math]*, Feb. 2020. arXiv: 2002.01696.

[43] S. Farazi, A. G. Klein, and D. Richard Brown, "Average age of information in multi-source self-preemptive status update systems with packet delivery errors," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 396–400, 2019.

[44] R. D. Yates and S. K. Kaul, "Age of Information in Uncoordinated Unslotted Updating," *arXiv:2002.02026 [cs, math]*, Feb. 2020. arXiv: 2002.02026.

[45] E. T. Ceran, D. Gündüz, and A. György, "Average Age of Information With Hybrid ARQ Under a Resource Constraint," *IEEE Transactions on Wireless Communications*, vol. 18, pp. 1900–1913, Mar. 2019.

[46] E. T. Ceran, D. Gündüz, and A. György, "A Reinforcement Learning Approach to Age of Information in Multi-User Networks," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1967–1971, Sept. 2018. ISSN: 2166-9589.

[47] E. Sert, C. Sönmez, S. Baghaee, and E. Uysal-Biyikoglu, "Optimizing age of information on real-life TCP/IP connections through reinforcement learning," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, May 2018.

[48] C. Kam, S. Kompella, and A. Ephremides, "Learning to sample a signal through an unknown system for minimum aoi," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 177–182, 2019.

[49] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *arXiv:1608.08622 [cs, math]*, Aug 2016.

[50] A. R. Teel, A. Subbaraman, and A. Sferlazza, "Stability analysis for stochastic hybrid systems: A survey," *Automatica*, vol. 50, pp. 2435–2456, Oct. 2014.

[51] J. Hespanha, "Modelling and analysis of stochastic hybrid systems," *IEE Proceedings - Control Theory and Applications*, vol. 153, pp. 520–535, Sept. 2006.

[52] R. D. Yates and S. Kaul, "Real-time status updating: Multiple sources," in *2012 IEEE International Symposium on Information Theory Proceedings*, pp. 2666–2670, July 2012. ISSN: 2157-8095.

[53] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Minimizing the Age of Information Through Queues," *IEEE Transactions on Information Theory*, vol. 65, pp. 5215–5232, Aug. 2019. Conference Name: IEEE Transactions on Information Theory.

[54] E. Najm and E. Telatar, "Status updates in a multi-stream M/G/1/1 preemptive queue," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, (Honolulu, HI), pp. 124–129, IEEE, Apr. 2018.

[55] L. Huang and E. Modiano, "Optimizing age-of-information in a multi-class queueing system," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 1681–1685, June 2015. ISSN: 2157-8117.

[56] M. Shaked and J. G. Shanthikumar, *Stochastic orders*. Springer series in statistics, New York: Springer, 2007.

[57] C.-D. Lai and M. Xie, *Stochastic ageing and dependence for reliability*. New York, NY: Springer Science + Business Media, 2006.

[58] O. Dogan and N. Akar, "The multi-source preemptive m/ph/1/1 queue with packet errors: Exact distribution of the age of information and its peak," 2020.

[59] A. M. Bedewy, Y. Sun, S. Kompella, and N. B. Shroff, "Age-optimal Sampling and Transmission Scheduling in Multi-Source Systems," *arXiv:1812.09463 [cs, math]*, Dec. 2018.

[60] C. J. Watkins and P. Dayan, "Q-Learning," *Machine Learning*, vol. 8, pp. 279–292, May 1992.

[61] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

APPENDIX A

TRANSITION TABLES OF SHS MODELS

l	$q_l \to q_l'$	$\lambda^{(l)}$	$\mathbf{x}\mathbf{A}_l$	$\mathbf{A}_l$	$\mathbf{v}_{q_l}\mathbf{A}_l$
1	$0 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ v_{00}\ 0]$
2	$0 \to 2$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ 0\ v_{01}]$
3	$1 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{10} - v_{11}\ v_{12}\ 0]$
4	$2 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{20} - v_{21}\ v_{22}\ 0]$
5	$1 \to 4$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ 0\ v_{11}]$
6	$4 \to 1$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{40} - v_{41}\ v_{42}\ 0]$
7	$3 \to 4$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{30}\ 0\ v_{31}]$
8	$4 \to 3$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{40}\ v_{40}\ 0]$
9	$3 \to 2$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{30} - v_{31}\ v_{32}\ 0]$
10	$2 \to 3$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{20}\ v_{20}\ 0]$
11	$4 \to 4$	λ_2	$[x_0\ x_1\ x_2]$	I	$[v_{40}\ v_{41}\ v_{42}]$
12	$1 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ v_{10}\ 0]$
13	$2 \to 2$	λ_2	$[x_0\ x_1\ x_2]$	I	$[v_{20}\ v_{21}\ v_{22}]$
14	$3 \to 3$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{30}\ v_{30}\ 0]$

Table A.1: The transition list of LCFS-Preemptive model given in Figure 3.1

l	$q_l \to q_l'$	$\lambda^{(l)}$	$\mathbf{x}\mathbf{A}_l$	$\mathbf{A}_l$	$\mathbf{v}_{q_l}\mathbf{A}_l$
1	$0 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ v_{00}\ 0]$
2	$1 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{10} - v_{11}\ v_{12}\ 0]$
3	$1 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ v_{10}\ 0]$
4	$2 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{20}\ v_{20}\ 0]$
5	$1 \to 2$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ 0\ v_{11}]$
6	$0 \to 2$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ 0\ v_{01}]$
7	$2 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{20} - v_{21}\ v_{22}\ 0]$
8	$0 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{00} - v_{01}\ v_{02}\ 0]$

Table A.2: The transition list of simplified version of LCFS-Preemptive model given in Figure 3.2

l	$q_l \to q_l'$	$\lambda^{(l)}$	$\mathbf{x}\mathbf{A}_l$	$\mathbf{A}_l$	$\mathbf{v}_{q_l}\mathbf{A}_l$
1	$1 \to 0$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ v_{10}\ 0]$
2	$0 \to 1$	λ_2	$[x_0\ 0\ x_1]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ 0\ v_{01}]$
3	$1 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{10} - v_{11}\ v_{12}\ 0]$
4	$0 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{00} - v_{01}\ v_{02}\ 0]$
5	$0 \to 0$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ v_{00}\ 0]$

Table A.3: The transition list of more simplified version of LCFS-Preemptive model given in Figure 3.3

l	$q_l \to q_l'$	$\lambda^{(l)}$	$\mathbf{x}\mathbf{A}_l$	$\mathbf{A}_l$	$\mathbf{v}_{q_l}\mathbf{A}_l$
1	$0 \to 1$	λ_1	$[x_0\ x_0\ 0]$	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ v_{00}\ 0]$
2	$0 \to 1$	λ_2	$[x_0\ 0\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{00}\ 0\ 0]$
3	$1 \to 0$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{10} - v_{11}\ v_{12}\ 0]$
4	$1 \to 2$	λ_1	$[x_0\ x_1\ x_0 - x_1]$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{10}\ v_{11}\ v_{10} - v_{11}]$
5	$1 \to 3$	λ_2	$[x_0\ x_1\ x_2]$	I	$[v_{10}\ v_{11}\ v_{12}]$
6	$2 \to 1$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{20} - v_{21}\ v_{22}\ 0]$
7	$2 \to 2$	λ_1	$[x_0\ x_1\ x_0 - x_1]$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{20}\ v_{21}\ v_{20} - v_{21}]$
8	$2 \to 4$	λ_2	$[x_0\ x_1\ x_2]$	I	$[v_{20}\ v_{21}\ v_{22}]$
9	$3 \to 1$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{30} - v_{31}\ v_{32}\ 0]$
10	$3 \to 5$	λ_1	$[x_0\ x_1\ x_0 - x_1]$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{30}\ v_{31}\ v_{30} - v_{31}]$
11	$4 \to 2$	μ	$[x_0 - x_1\ 0\ x_2]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$[v_{30} - v_{31}\ 0\ v_{32}]$
12	$4 \to 5$	λ_1	$[x_0\ x_1\ x_0 - x_1]$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{40}\ v_{41}\ v_{40} - v_{41}]$
13	$5 \to 3$	μ	$[x_0 - x_1\ x_2\ 0]$	$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$[v_{50} - v_{51}\ v_{52}\ 0]$
14	$5 \to 5$	λ_1	$[x_0\ x_1\ x_0 - x_1]$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$[v_{50}\ v_{51}\ v_{50} - v_{51}]$
15	$5 \to 4$	λ_2	$[x_0\ x_1\ x_2]$	I	$[v_{50}\ v_{51}\ v_{52}]$

Table A.4: The transition list of LCFS-Nonpreemptive model given in Figure 3.5