2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)

Designing Approximate and Deployable SRPT Scheduler: A Unified Framework

Zhiyuan Wang*, Jiancheng Ye†, Dong Lin†, Yipei Chen†, and John C.S. Lui*
*Department of Computer Science and Engineering, The Chinese University of Hong Kong
†Network Technology Lab and Hong Kong Research Center, Huawei Technologies Co., Ltd.
Email: {zywang, cslui}@cse.cuhk.edu.hk, {yejiancheng, lin.dong, chen.yipei}@huawei.com

2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS) | 978-1-6654-1494-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/IWQOS52092.2021.9521259

***Abstract*—The scheduling policy installed on switches of datacenters plays a significant role on congestion control. Shortest-Remaining-Processing-Time (SRPT) achieves the near-optimal average message completion time (MCT) in various scenarios, but is difficult to deploy as viewed by the industry. The reasons are two-fold: 1) many commodity switches only provide FIFO queues, and 2) the information of remaining message size is not available. Recently, the idea of emulating SRPT using only a few FIFO queues and the original message size has been coined as the approximate and deployable SRPT (ADS) design. In this paper, we provide the first theoretical study on ADS design. Specifically, we first characterize a wide range of feasible ADS scheduling policies via a unified framework, and then derive the steady-state MCT and slowdown in the M/G/1 setting. We formulate the optimal ADS design as a non-linear combinatorial optimization problem, which aims to minimize the average MCT given the available FIFO queues. To prevent the starvation of long messages, we also take into account the fairness condition based on the steady-state slowdown. The optimal ADS design problem is NP-hard in general, and does not exhibit monotonicity or sub-modularity. We leverage its decomposable structure and devise an efficient algorithm to solve the optimal ADS policy. Numerical results based on the realistic heavy-tail message size distribution show that the optimal ADS policy installed on eight FIFO queues is capable of emulating the true SRPT in terms of MCT and slowdown.**

I. Introduction

A. Background and Motivation

Congestion control (CC) is one of the most critical issues in the modern data-center network (DCN). To maintain reliability and scalability, most CC schemes coordinate in a distributed way, and is partly implemented in network switches [1]. Hence the end-to-end latency depends primarily on the scheduling policy installed on the switches. Nowadays, many datacenter applications are using request-response protocols, which generate a lot of short messages. The application message is a block of packets transmitted from a sender to the receiver. In general, the message-size-based scheduling policy that prioritizes the short messages is believed to significantly reduce the average message completion time (MCT). Moreover, Shortest-Remaining-Processing-Time (SRPT) is known to achieve the near-optimal average completion time [2], by prioritizing jobs with the shortest *remaining* service time. However, SRPT is not readily deployable as viewed by the industry for two reasons:

This work is supported in part by the GRF 14201819 and CUHK: 6905407.

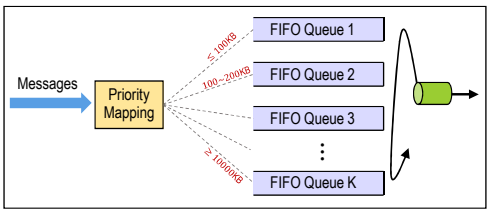


Fig. 1: An illustrative ADS design on K FIFO queues

- First, many commodity switches by default only provide support for FIFO queues per outgoing port, thus naturally enforce the First-Come-First-Serve (FCFS) discipline.
- Second, the *remaining* message size information needed for SRPT is not available in today's transport protocols.

There have been some studies (e.g., [3]–[8]) on how to harness the advantage of SRPT via feasible industry solutions. A promising approach is to emulate SRPT based on the preemptive size-based scheduling policies. The feasibility of this idea relies on the following two facts.

- First, most commodity switches provide support for *multiple* FIFO queues (typically eight to ten [3], [9]).
- Second, the *original* size of a message is available to the switch.[1] This information allows the switch to prioritize the shorter messages in transmission.

The above two facts make it possible to emulate SRPT by putting shorter messages into the FIFO queue with a higher transmission priority, so that they can finish faster. This idea is coined as ***approximate and deployable SRPT (ADS)*** by Mushtaq *et al.* in [8]. Fig. 1 provides an illustration based on the switch with K FIFO queues. It works as follows:

- The switch assigns the incoming message based on its original message sizes to one of the K FIFO queues. Each FIFO queue has a predefined size interval. Here, a smaller queue index means a higher priority for transmission.
- The switch will transmit the messages based on the predefined priority. A newly-arrived short message (e.g., assigned to FIFO queue 2) can preempt the long message being transmitted (e.g., in FIFO queue 3).

Mushtaq *et al.* in [8] explore the ADS design space in different dimensions based on the packet-level simulation (see

[1]RDMA is completely message-orientated [10]. The sender must specify the size information in the first packet to be transmitted [6].

Section 3.1 in [8]). Our work in this paper provides the first theoretical study on two critical dimensions design in ADS, i.e., the priority mapping and the number of used FIFO queues. Despite some system-level ADS studies (e.g., [3]–[8]), there is no theoretical framework on how to jointly optimize the size-based priority mapping and the number of adopted priority levels. In this paper, we take the initial step to fill this void, and propose a unified theoretical framework for designing ADS scheduling policy. Specifically, we will address the following two fundamental questions:

Question 1. *Given $K \geq 2$ FIFO queues in the switch, what is the optimal size-based priority mapping scheme?*

Question 2. *If the switch could provide sufficient FIFO queues, what is the optimal number of priority levels?*

Question 1 corresponds to the practical ADS scheduling policy design given the switch with a fixed number of FIFO queues. For Question 2, we investigate the optimal number of priority levels to adopt. Although more available FIFO queues provide more design flexibility, we show that the optimal ADS scheduling policy is *not necessary* to utilize all the available FIFO queues. The number of used FIFO queues may depend on factors such as the message size distribution and the network load intensity. By investigating Question 2, we will reveal when it is necessary to upgrade the current switch by increasing the embedded FIFO queues.

B. Main Results and Key Contributions

Our main results and key contributions in this paper are summarized as follows:

- *A Unified Framework for ADS Scheduler:* We leverage the discrete message size, and characterize a wide range of ADS scheduling policies via a unified framework. The scheduling policies within this framework only require the original message size and a few FIFO queues, thus provide a feasible industry solution for commodity switches. Under this framework, we derive the steady-state MCT and slowdown in the M/G/1 setting, which facilitates the subsequent optimal ADS policy design.
- *Problem Formulation for ADS Policy Design:* We formulate the optimal ADS design as a non-linear combinatorial optimization problem, with the goal to minimize the average MCT given the available FIFO queues in the switch. To avoid the starvation of long messages, we also take into account the fairness condition based on the maximal slowdown among different sizes of messages.
- *Optimal ADS Policy:* The above ADS design problem is NP-hard in general, and does not exhibit monotonicity or sub-modularity. We leverage the decomposable structure in this problem, and devise an efficient algorithm to solve the optimal ADS policy. Our approach leads to the optimal priority mapping, and also unveils the optimal number of priority levels. As far as we know, we are the first to jointly address the two fundamental challenges in ADS design.
- *Performance Evaluation:* Numerical results based on the realistic heavy-tail message size distribution show that the optimal ADS policy on eight FIFO queues can maintain almost the same performance as `SRPT` in the M/G/1 setting.

The remainder of this paper is organized as follows. Section II introduces the system model and the ADS design problem. Section III derives the optimal ADS policy. Section IV provides numerical results. Section V concludes this paper.

II. System Model

We study the ADS scheduling policy design installed on the switch of DCN, and model this problem based on the M/G/1 queueing system. The messages of different sizes arrive at the switch according to a Poisson process with the rate λ. We quantify the message size in the number of packets, and model it as a discrete random variable on the support set $\mathcal{N} = \{1, 2, ..., N\}$, where N indicates the maximal message size. Let $f(n)$ denote the probability mass function (PMF), and let $F(n) \triangleq \sum_{i=1}^{n} f(i)$ denote the cumulative distribution function (CDF). Without loss of generality, we consider a normalized bandwidth, thus the service time of transmitting a size-n message is n. Accordingly, we follow some classic notations from [11] and denote the expected service time as

$$\frac{1}{\mu} \triangleq \sum_{n=1}^{N} n f(n), \tag{1}$$

where μ is the serving rate, and the network load is $\rho \triangleq \frac{\lambda}{\mu}$.

Next we characterize the ADS scheduling policy via a unified framework in Section II-A. We then introduce the priority mapping and the steady-state performance of the ADS policy in Section II-B and Section II-C, respectively. We formulate the ADS design problem in Section II-D.

A. Unified Framework for ADS Policy

We focus our ADS design on preemptive size-based scheduling, and propose a unified framework in the following.

1) General Characterization: We use the binary variable $x_n \in \{0, 1\}$ to denote the preemption design associated with the message size $n \in \mathcal{N}$. The physical meaning is as follows:

- The case of $x_n = 0$ represents that the size-n messages have the same priority as the messages of size $n + 1$. In this case, any message of size n and size $n + 1$ will be assigned to the same FIFO queue in the switch, thus will be transmitted according to the `FCFS` discipline.
- The case of $x_n = 1$ represents that message size n has a higher priority than other message sizes greater than n. It has two-fold implications. First, any size-n message will be assigned to FIFO queues with a higher priority than the messages greater than n. Second, a newly-arrived size-n message can immediately preempt the current transmission of a message greater than n in the switch.

Based on the above discussion, we characterize a wide range of preemptive size-based scheduling policies as the following N-dimensional binary vector:

$$\boldsymbol{x} \triangleq (x_n \in \{0, 1\} : \forall n \in \mathcal{N}). \tag{2}$$

We say that the message size $n \in \mathcal{N}$ is a *preemption point* if and only if $x_n = 1$. Note that $x_N \in \{0, 1\}$ has no effect on the scheduling policy $\boldsymbol{x}$, since N is the maximal message size. For notation simplicity, we will fix $x_N = 1$. Accordingly, the ADS policy $\boldsymbol{x}$ is chosen from the policy set $\mathcal{X}$, i.e.,

$$\mathcal{X} \triangleq \left(\boldsymbol{x} \in \{0,1\}^N : \ x_N = 1\right). \tag{3}$$

2) Special Cases: The vector $\boldsymbol{x} \in \mathcal{X}$ generalizes two well-known scheduling policies as the special cases.

- The case of $\boldsymbol{x} = \mathbf{1}_N$ corresponds to Preemptive-Shortest-Job-First (PSJF), where $\mathbf{1}_N$ is an N-dimensional all-one vector. It strictly prioritizes the messages of the smallest original size. Given the message size set $\mathcal{N}$, it requires N FIFO queues to implement PSJF on the switch, one for each message size. Hence PSJF is costly.
- The case of $\boldsymbol{x} = (\mathbf{0}_{N-1}, 1)$ corresponds to First-Come-First-Serve (FCFS), which has no prioritization across all the message sizes (i.e., $x_N = 1$ has no effect). It only requires one FIFO queue to implement FCFS on the switch.

Note that PSJF requires more FIFO queues than FCFS, and also attains a smaller average MCT. In the M/G/1 scenario, PSJF is known to be 1.5-competitive to SRPT in terms of the average MCT [12]. Nevertheless, we will show later that it is possible to devise an ADS policy that achieves an even smaller average MCT using much fewer FIFO queues than PSJF.

B. Message Assignment to FIFO Queues

The scheduling priority defined by the vector $\boldsymbol{x} \in \mathcal{X}$ determines a unique message assignment to the FIFO queues. We use a simple example to illustrate the connection between the binary vector $\boldsymbol{x}$ and the FIFO queues.

Example 1. *Suppose that $N = 9$, then the policy $\boldsymbol{x} = (0, 0, 1, 0, 0, 0, 1, 0, 1)$ requires a total of three FIFO queues.*

- *The messages of sizes $\{1, 2, 3\}$ will be assigned to FIFO queue 1, which has the highest priority.*
- *The messages of sizes $\{4, 5, 6, 7\}$ will be assigned to FIFO queue 2, which has the second highest priority.*
- *The messages of sizes $\{8, 9\}$ will be assigned to FIFO queue 3, which has the lowest priority.*

The above example shows that the ADS policy $\boldsymbol{x} \in \mathcal{X}$ requires a total of $\sum_{n=1}^{N} x_n$ FIFO queues. To analyze the message assignment outcome, we need to define some intermediate notations. We let $l_n(\boldsymbol{x})$ denote the largest preemption point in the message size subset $\{1, 2, ..., n-1\}$, i.e.,

$$l_n(\boldsymbol{x}) \triangleq \max_{1 \le i < n} \ i \cdot x_i \qquad \textit{s.t. } x_i = 1. \tag{4}$$

Similarly, we let $r_n(\boldsymbol{x})$ denote the smallest preemption point in the message size subset $\{n, n+1, ..., N\}$, i.e.,

$$r_n(\boldsymbol{x}) \triangleq \min_{n \le i \le N} \ i \cdot x_i \qquad \textit{s.t. } x_i = 1. \tag{5}$$

Table I illustrates $l_n(\boldsymbol{x})$ and $r_n(\boldsymbol{x})$ based on Example 1. Next, we present the message assignment outcome of the policy $\boldsymbol{x} \in \mathcal{X}$ based on $l_n(\boldsymbol{x})$ and $r_n(\boldsymbol{x})$ in Proposition 1.

Proposition 1. *For any $n \in \mathcal{N}$, the messages of sizes in the set $\{l_n(\boldsymbol{x}) + 1, l_n(\boldsymbol{x}) + 2, ..., r_n(\boldsymbol{x})\}$ will be assigned to the FIFO queue with the $K_n(\boldsymbol{x})$-th priority, where $K_n(\boldsymbol{x})$ is*

$$K_n(\boldsymbol{x}) \triangleq \sum_{i=1}^{r_n(\boldsymbol{x})} x_i. \tag{6}$$

Next we will derive the steady-state performance based on the message assignment outcome in Proposition 1.

TABLE I: An illustration based on Example 1

Message Size n	1	2	3	4	5	6	7	8	9
$\boldsymbol{x} = (x_n : \forall n \in \mathcal{N})$	0	0	1	0	0	0	1	0	1
$l_n(\boldsymbol{x})$	0	0	0	3	3	3	3	7	7
$r_n(\boldsymbol{x})$	3	3	3	7	7	7	7	9	9

C. Steady-State Performance

1) Performance Metric: We focus on two performance metrics: message completion time (MCT) and slowdown. The MCT measures the time from when the first packet of a message is sent until the last packet is received. When we focus on a single switch, MCT is equivalent to the response time in queueing system. We let $T_n(\boldsymbol{x})$ denote the *steady-state* MCT of the size-n message under ADS policy $\boldsymbol{x}$. The average steady-state MCT achieved by the policy $\boldsymbol{x} \in \mathcal{X}$ is

$$\bar{T}(\boldsymbol{x}) \triangleq \sum_{n=1}^{N} T_n(\boldsymbol{x}) f(n). \tag{7}$$

The slowdown of a message is the weighted response time measure, which is defined as the response time of the message divided by the time that the same message would take to complete if it was the only message in the system (e.g., [3]–[6]). Given the normalized bandwidth, the steady-state slowdown seen by the size-n message is

$$S_n(\boldsymbol{x}) \triangleq \frac{T_n(\boldsymbol{x})}{n}. \tag{8}$$

We will derive the steady-state MCT $T_n(\boldsymbol{x})$ based on a series of auxiliary systems to be defined in the following.

2) Auxiliary System: Based on the network workload $\{\lambda, \mathcal{N}, f(\cdot)\}$, we define the auxiliary system $\mathcal{A}(i)$ as follows:

Definition 1. *The auxiliary system $\mathcal{A}(i)$ is an M/G/1 queueing system satisfying the following conditions:*

1) The arrival rate of $\mathcal{A}(i)$ is $\lambda_A \triangleq \lambda F(i)$.

2) The service time distribution of $\mathcal{A}(i)$ is

$$f_A(t) \triangleq \begin{cases} f(t)/F(i), & \textit{if } t \in \{1, 2, ..., i\}, \\ 0, & \textit{otherwise}. \end{cases} \tag{9}$$

The steady-state analysis of policy $\boldsymbol{x} \in \mathcal{X}$ is closely related to the auxiliary systems. To facilitate the later discussion, we follow some classic notations from [11], and introduce three formulas $\rho(i)$, $V(i)$, and $W(i)$ for the auxiliary system $\mathcal{A}(i)$. First, let $\rho(i)$ denote the load of auxiliary system $\mathcal{A}(i)$, i.e.,

$$\rho(i) \triangleq \lambda_A \sum_{t=1}^{i} t f_A(t) = \lambda \sum_{t=1}^{i} t f(t). \tag{10}$$

Second, we let $V(i)$ denote the expected remaining service time of the job being served at a random time point in the auxiliary system $\mathcal{A}(i)$. In the M/G/1 model, we have

$$V(i) \triangleq \frac{\lambda_A}{2}\sum_{t=1}^{i} t^2 f_A(t) = \frac{\lambda}{2}\sum_{t=1}^{i} t^2 f(t). \tag{11}$$

Third, we let $W(i)$ denote the expected remaining service time of the jobs in the auxiliary system $\mathcal{A}(i)$ at a random time point. Based on Kleinrock's Conservation Law [13], we have

$$W(i) \triangleq \frac{V(i)}{1-\rho(i)}. \tag{12}$$

3) Steady-State Analysis: We take the representative size-n message as an example, and derive the steady-state MCT $T_n(\boldsymbol{x})$ based on the aforementioned auxiliary systems. The steady-state MCT $T_n(\boldsymbol{x})$ satisfies the following equation:

$$T_n(\boldsymbol{x}) = \underbrace{W\big(r_n(\boldsymbol{x})\big)}_{\text{Part I}} + \underbrace{n}_{\text{Part II}} + \underbrace{T_n(\boldsymbol{x})\rho\big(l_n(\boldsymbol{x})\big)}_{\text{Part III}}, \tag{13}$$

which implies that the steady-state MCT $T_n(\boldsymbol{x})$ equals to the summation of three parts. Next, we elaborate the physical meaning of the three parts on the right-hand-side of (13):

- Part I in (13) represents the expected remaining service time of messages in the top $K_n(\boldsymbol{x})$ FIFO queues when the size-n message arrives.
- Part II in (13) represents the service time of transmitting the size-n message itself given the normalized bandwidth.
- Part III in (13) represents the expected service time of the messages that arrive later but have a higher priority than the size-n message during the completion time $T_n(\boldsymbol{x})$.

Based on Equation (13), we derive the steady-state MCT of the size-n message as follows:

$$T_n(\boldsymbol{x}) = \left[\frac{V(r_n(\boldsymbol{x}))}{1-\rho(r_n(\boldsymbol{x}))} + n\right]\frac{1}{1-\rho(l_n(\boldsymbol{x}))}. \tag{14}$$

D. Problem Formulation

The goal of ADS design is to emulate `SRPT`, thus we aim to minimize the average steady-state MCT defined in (7). We also need to take into account the following two aspects:

- First, let K denote the number of available FIFO queues in the switch, which is the hardware limitation in practice. Hence a feasible ADS policy $\boldsymbol{x} \in \mathcal{X}$ should not require more than K FIFO queues, i.e.,

$$\sum_{n=1}^{N} x_n \le K. \tag{15}$$

- Second, we take into account the fairness issue in ADS design. Recall that reducing the average MCT requires that the scheduling policy should prioritize the short messages. To prevent the starvation of long messages, we introduce the following constraints

$$S_n(\boldsymbol{x}) \le S_{\max}, \quad \forall n \in \mathcal{N}, \tag{16}$$

which indicates that the *maximal steady-state slowdown* should be no greater than $S_{\max}$. One can flexibly choose the parameter $S_{\max}$ in practice. According to the previous studies (e.g., [14]), the proportional fairness criteria corresponds to $S_{\max} = \frac{1}{1-\rho}$, where $\rho = \frac{\lambda}{\mu}$ is the load.

Based on the above discussions, we formulate the optimal ADS policy design in Problem 1, which aims to minimize the average MCT considering the above two critical aspects.

Problem 1 (ADS Policy Design)**.**

$$\begin{aligned} \boldsymbol{x}^\star \triangleq \arg\min \quad & \sum_{n=1}^{N} T_n(\boldsymbol{x}) f(n) \\ \textit{s.t.} \quad & (15), (16), \\ \textit{var.} \quad & \boldsymbol{x} \in \mathcal{X}. \end{aligned} \tag{17}$$

Problem 1 is a non-linear combinatorial optimization problem, which is NP-hard in general. It does not exhibit monotonicity or sub-modularity, thus the greedy algorithm has no performance guarantee here. In Section III, we will introduce how to efficiently solve it using its decomposable structure.

III. ADS Policy Design

We first introduce the key idea of solving the optimal ADS policy $\boldsymbol{x}^\star$ in Problem 1, and then present the algorithm.

A. Key Idea of Solving Problem 1

1) Decomposable Structure: The definition of the N-dimensional binary vector $\boldsymbol{x}$ endows Problem 1 with a decomposable structure. Specifically, the preemption design $x_n = 1$ indicates that the messages of sizes $\{1, 2, ..., n\}$ have a higher priority than the messages of sizes $\{n+1, n+2, ..., N\}$. Mathematically, $x_n = 1$ implies that the steady-state MCT $T_i(\boldsymbol{x})$ will not be affected by x_j for any $i, j \in \mathcal{N}$ satisfying $i \le n < j$. Hence the preemption design $x_n = 1$ enables us to decompose the objective function in Problem 1 with respect to the size index n. This allows us to decompose Problem 1 into sub-problems, and then solve the original problem in a *recursive manner*. Next we define the sub-problem.

2) Sub-Problem for ADS Design: Based on Problem 1, we will define a series of sub-problems for ADS policy design. Problem 2 corresponds to the type-(k, i) sub-problem.

Problem 2. *The type-(k, i) sub-problem is*

$$\begin{aligned} H(k,i) \triangleq \min \quad & \sum_{n=1}^{i} T_n(\boldsymbol{x}) f(n) && \text{(18a)} \\ \textit{s.t.} \quad & x_i = 1, && \text{(18b)} \\ & \sum_{n=1}^{i} x_n \le k, && \text{(18c)} \\ & S_n(\boldsymbol{x}) \le S_{\max}, \ \forall n \le i, && \text{(18d)} \\ \textit{var.} \quad & \{x_1, x_2, ..., x_i\} \in \{0,1\}^i. && \text{(18e)} \end{aligned}$$

We let $H(k, i)$ denote the optimal value of the type-(k, i) sub-problem. For presentation convenience, we will refer to

$H(\cdot,\cdot)$ as the cost matrix. Note that the type-(K,N) sub-problem is mathematically equivalent to Problem 1. Therefore, the optimal ADS policy $\boldsymbol{x}^\star$ and the cost $H(K,N)$ satisfy

$$H(K,N)=\sum_{n=1}^{N}T_n(\boldsymbol{x}^\star)f(n). \tag{19}$$

In Section III-B, we will derive $\boldsymbol{x}^\star$ based on the cost matrix $H(\cdot,\cdot)$. We first introduce how to efficiently calculate the cost matrix according to the following recursive relation.

3) Recursive Relation: To facilitate the later discussion, we define two intermediate functions as follows:

$$\hat{T}_n(s,e)=\left[\frac{V(e)}{1-\rho(e)}+n\right]\frac{1}{1-\rho(s)}, \tag{20a}$$

$$\hat{S}_n(s,e)=\left[\frac{V(e)}{1-\rho(e)}\cdot\frac{1}{n}+1\right]\frac{1}{1-\rho(s)}, \tag{20b}$$

where $\rho(\cdot)$ and $V(\cdot)$ are given in (10) and (11), respectively. Based on (20), one can express the size-n message's steady-state MCT and slowdown as follows:

$$\begin{aligned}T_n(\boldsymbol{x})&=\hat{T}_n\big(l_n(\boldsymbol{x}),r_n(\boldsymbol{x})\big),\\ S_n(\boldsymbol{x})&=\hat{S}_n\big(l_n(\boldsymbol{x}),r_n(\boldsymbol{x})\big).\end{aligned} \tag{21}$$

Lemma 1 presents the recursive relation for the cost matrix.

Lemma 1. *The cost $H(k,i)$ satisfies*

$$H(k,i)=\min_{0\le q<i}\quad H(k-1,q)+\sum_{n=q+1}^{i}\hat{T}_n(q,i)f(n) \tag{22a}$$

$$\textit{s.t.}\quad \hat{S}_{q+1}(q,i)\le S_{\max}. \tag{22b}$$

The recursive relation in Lemma 1 enables us to efficiently calculate the cost matrix $H(\cdot,\cdot)$ and derive the optimal ADS policy $\boldsymbol{x}^\star$. Next, we describe the details.

B. Algorithm Design

Algorithm 1 summarizes our proposed approach for solving the optimal ADS policy $\boldsymbol{x}^\star$. It works in two steps as follows:

- *Lines 2∼10* calculate the cost matrix $H(\cdot,\cdot)$ and the policy matrix $P(\cdot,\cdot)$. In the iteration (k,i), we utilize the recursive relation (22) to calculate $H(k,i)$ and $P(k,i)$ based on $\{H(k-1,q):\forall 0\le q<i\}$ recorded in previous iterations. Finally, we obtain the cost $H(k,i)=\Psi(q^*)$ and the policy $P(k,i)=q^*$ in Line 10.
- *Lines 11∼15* generate the optimal ADS policy $\boldsymbol{x}^\star$ based on the policy matrix $P(\cdot,\cdot)$. Specifically, we start with the type-(K,N) sub-problem in Line 11, and find out the optimal preemption points repetitively in Lines 12∼15.

The running time complexity and the space complexity of Algorithm 1 are $\mathcal{O}(KN^2)$ and $\mathcal{O}(KN)$, respectively. Note that our above analysis takes into account the limitation from the number of available FIFO queues, i.e., the constraint (15). Hence we have addressed Question 1. To obtain the optimal number of priority levels, i.e., Question 2, we can remove this constraint and investigate the optimal ADS design. The absence of constraint (15) in fact simplifies the sub-problem and the corresponding algorithm. It is not difficult to show that the corresponding algorithm has the running time complexity $\mathcal{O}(N^2)$ and the space complexity $\mathcal{O}(N)$.

Algorithm 1:

Input : Arrival rate λ and message size PMF $f(\cdot)$
Output: Optimal ADS policy $\boldsymbol{x}^\star$

1 **Initial** $H(0,0)=0$, $P(0,0)=0$, and $\boldsymbol{x}^\star=(\mathbf{0}_{N-1},1)$
2 **for** $k=1$ **to** K **do**
3 **for** $i=1$ **to** N **do**
4 **for** $q=0$ **to** $i-1$ **do**
5 **if** $\hat{S}_{q+1}(q,i)>S_{\max}$ **then**
6 $\Psi(q)=+\infty$
7 **else**
8 $\Psi(q)=H(k-1,q)+\sum_{n=q+1}^{i}\hat{T}_n(q,i)f(n)$
9 Find $q^*=\min_{0\le q<i}\Psi(q)$
10 Set $H(k,i)=\Psi(q^*)$ and $P(k,i)=q^*$
11 Set $k=K$ and $n=N$
12 **repeat**
13 Set $n=P(k,n)$ and $x_n^\star=1$
14 $k=k-1$
15 **until** $k=0$;

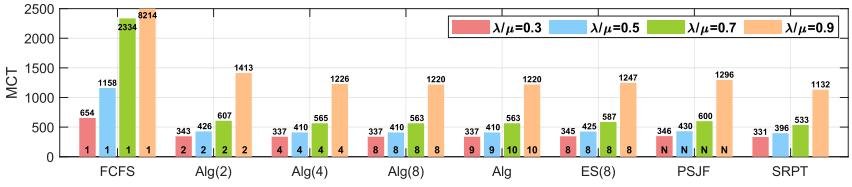
IV. Numerical Results

We will evaluate the performance of the optimal ADS design, and compare the steady-state performance of different scheduling policies in the M/G/1 setting. We follow the previous studies (e.g., [3], [8], [15]) and assume that the message size follows a realistic heavy-tail distribution, where 50% of the messages are of 1KB, 35% are between (1KB,201KB], and 15% are between (201KB,3000KB]. Hence we have $N=3000$. We take `FCFS`, `PSJF`, and `SRPT` as benchmarks, and evaluate the following cases:

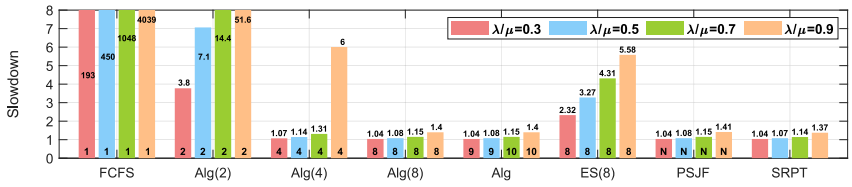
- Case `Alg`(K) denotes the optimal ADS policy $\boldsymbol{x}^\star$ generated by Algorithm 1, which requires at most K FIFO queues due to the constraint (15).
- Case `Alg` denotes the optimal ADS policy without the constraint (15). Hence `Alg` may require up to N FIFO queues, which is unknown in advance.
- Case `ES`(K) denotes the simple equal-splitting heuristic installed on K FIFO queues, which is used in some previous studies (e.g., [3], [4], [8]).

Fig. 2 plots the average steady-state MCT and slowdown. In each sub-figure, the four bars for each scheduling policy represent different load $\lambda/\mu\in\{0.3,0.5,0.7,0.9\}$. At the bottom of each bar, we label the number of required FIFO queues. At the top of each bar, we label the average MCT or slowdown. Fig. 2 leads to the following observations.

Observation 1: In each sub-figure of Fig. 2, comparing `Alg`(2) to `FCFS` unveils the significant improvement of the priority-based scheduling. The average MCT of `Alg`(K) slightly decreases in $K\in\{2,4,8\}$ as shown in Fig. 2(a), while the

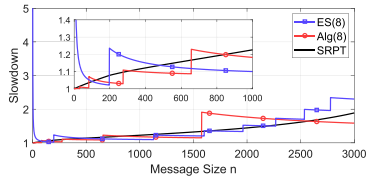


(a) Average MCT

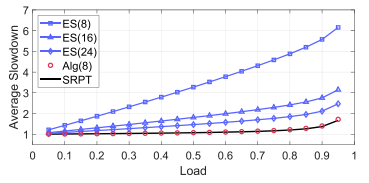


(b) Average slowdown

Fig. 2: Steady-state performance



(a) Slowdown at load 0.5



(b) $\texttt{Alg}(8)$ vs $\texttt{ES}(K)$ with $K \in \{8, 16, 24\}$

Fig. 3: Compare $\texttt{ES}(K)$ and $\texttt{Alg}(K)$

average slowdown of $\texttt{Alg}(K)$ has a significant drop from $K = 2$ to $K = 4$ as shown in Fig. 2(b). This is because that the average MCT is dominated by the contribution from just a few long messages, thus is not an informative metric when there are many short messages.

Observation 2: In each sub-figure of Fig. 2, comparing $\texttt{Alg}(8)$ to SRPT shows that the optimal ADS policy installed on eight FIFO queues is very close to the true SRPT in terms of the average MCT and slowdown. In particularly, $\texttt{Alg}(8)$ also outperforms PSJF (that requires 3000 FIFO queues) in terms of the average MCT and slowdown.

Observation 3: In Fig. 2(b), we find that $\texttt{Alg}(8)$ significantly outperforms $\texttt{ES}(8)$ in terms of the average slowdown. To get a better understanding, we further plot the steady-state slowdown for each message size $n \in \mathcal{N}$ under load 0.5 in Fig. 3(a), where the three curves correspond to SRPT, $\texttt{Alg}(8)$ and $\texttt{ES}(8)$, respectively. Fig. 3(a) shows that $\texttt{Alg}(8)$ performs much better than $\texttt{ES}(8)$ in terms of emulating SRPT for short messages (e.g., smaller than 200KB). Moreover, Fig. 3(b) compares the average slowdown of $\texttt{Alg}(8)$ and $\texttt{ES}(K)$ under different load. It shows that the equal-splitting heuristic under 24 FIFO queues is still not as good as $\texttt{Alg}(8)$.

Observation 4: Fig. 2 shows that the optimal number of used priority levels for case Alg are $\{9, 9, 10, 10\}$ given the load $\{0.3, 0.5, 0.7, 0.9\}$, respectively. This has two-fold implications. First, the optimal number of priority levels is much smaller than the number of different message sizes $N = 3000$. Second, a heavier load may lead to more priority levels in the optimal ADS design.

V. Conclusion

This paper studies approximate and deployable SRPT (ADS) design for switches in DCN. The ADS design aims to emulate SRPT relying only on a few FIFO queues and the original message size information. We first characterize a wide range of ADS policies via a unified framework as a binary vector, and then derive the steady-state performance in the M/G/1 setting. We formulate the optimal ADS policy design as a non-linear combinatorial optimization problem, which is NP-hard and does not exhibit monotonicity and sub-modularity. We leverage its decomposable structure, and devise an efficient algorithm to solve the optimal ADS policy.

References

[1] M. Noormohammadpour and C. S. Raghavendra, "Datacenter traffic control: Understanding techniques and tradeoffs," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1492–1525, 2017.

[2] D. R. Smith, "A new proof of the optimality of the shortest remaining processing time discipline," *Operations Research*, vol. 26, no. 1, pp. 197–199, 1978.

[3] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pfabric: Minimal near-optimal datacenter transport," *ACM SIGCOMM*, vol. 43, no. 4, pp. 435–446, 2013.

[4] W. Bai, L. Chen, K. Chen, D. Han, C. Tian, and H. Wang, "Information-agnostic flow scheduling for commodity data centers," in *NSDI*, 2015.

[5] P. X. Gao, A. Narayan, G. Kumar, R. Agarwal, S. Ratnasamy, and S. Shenker, "phost: Distributed near-optimal datacenter transport over commodity network fabric," in *ACM SIGCOMM*, 2015.

[6] B. Montazeri, Y. Li, M. Alizadeh, and J. Ousterhout, "Homa: A receiver-driven low-latency transport protocol using network priorities," in *ACM SIGCOMM*, 2018.

[7] Y. Lu, G. Chen, L. Luo, K. Tan, Y. Xiong, X. Wang, and E. Chen, "One more queue is enough: Minimizing flow completion time with explicit priority notification," in *IEEE INFOCOM*, 2017.

[8] A. Mushtaq, R. Mittal, J. McCauley, M. Alizadeh, S. Ratnasamy, and S. Shenker, "Datacenter congestion control: Identifying what is essential and making it practical," *ACM SIGCOMM Computer Communication Review*, 2019.

[9] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," in *ACM SIGCOMM*, 2010.

[10] P. MacArthur and R. D. Russell, "A performance study to guide rdma programming decisions," in *IEEE HPCC*, 2012.

[11] T. O'Donovan, "Direct solutions of m/g/1 priority queueing models," *RAIRO-Operations Research-Recherche Opérationnelle*, vol. 10, no. V1, pp. 107–111, 1976.

[12] A. Wierman, "Scheduling for today's computer systems: Bridging theory and practice," Ph.D. dissertation, Carnegie Mellon University, 2007.

[13] L. Kleinrock, "A conservation law for a wide class of queueing disciplines," *Naval Research Logistics Quarterly*, vol. 12, no. 2, pp. 181–192, 1965.

[14] A. Wierman, "Fairness and scheduling in single server queues," *Surveys in Operations Research and Management Science*, vol. 16, no. 1, pp. 39–48, 2011.

[15] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *ACM SIGCOMM*, 2010.