Review on Service Curves of Typical Scheduling Algorithms

GAO Yuehong[1], NING Zhi[1], HE Jia[1], ZHOU Jinfei[1], GAO Chenqiang[2,3], TANG Qingkun[2,3], YU Jinghai[2,3]

(1. Beijing University of Posts and Telecommunications, Beijing 100876, China;
2. ZTE Corporation, Shenzhen 518057, China;
3. State Key Laboratory of Mobile Network and Mobile Multimedia Technology, Shenzhen 518055, China)

DOI: 10.12142/ZTECOM.202402008

https://kns.cnki.net/kcms/detail/34.1294.TN.20240516.1410.002.html, published online May 16, 2024

Manuscript received: 2024-01-03

Abstract: In recent years, various internet architectures, such as Integrated Services (IntServ), Differentiated Services (DiffServ), Time Sensitive Networking (TSN) and Deterministic Networking (DetNet), have been proposed to meet the quality-of-service (QoS) requirements of different network services. Concurrently, network calculus has found widespread application in network modeling and QoS analysis. Network calculus abstracts the details of how nodes or networks process data packets using the concept of service curves. This paper summarizes the service curves for typical scheduling algorithms, including Strict Priority (SP), Round Robin (RR), Cycling Queuing and Forwarding (CQF), Time Aware Shaper (TAS), Credit Based Shaper (CBS), and Asynchronous Traffic Shaper (ATS). It introduces the theory of network calculus and then provides an overview of various scheduling algorithms and their associated service curves. The delay bound analysis for different scheduling algorithms in specific scenarios is also conducted for more insights.

Keywords: network calculus; service curve; scheduling algorithm; QoS

Citation (Format 1): GAO Y H, NING Z, HE J, et al. Review on service curves of typical scheduling algorithms [J]. ZTE Communications, 2024, 22(2): 55 - 70. DOI: 10.12142/ZTECOM.202402008

Citation (Format 2): Y. H. Gao, Z. Ning, J. He, et al., "Research on multi-core processor analysis for WCET estimation," *ZTE Communications*, vol. 22, no. 2, pp. 55 - 70, Jun. 2024. doi: 10.12142/ZTECOM.202402008.

1 Introduction

With the rapid advancement of internet technology and applications, the variety of services within networks has continuously expanded and network structure has grown increasingly complex. The traditional internet, which relies on the best effort (BE) service model and the First Come First Served (FCFS) scheduling algorithm, can no longer meet the diverse quality-of-service (QoS) requirements of different services. To ensure QoS, international standardization organizations have introduced several Internet architectures, including Integrated Services (IntServ), Differentiated Services (DiffServ), Time-Sensitive Networking (TSN), and Deterministic Networking (DetNet).

The IntServ model reserves network resources based on the QoS requirements of a given traffic flow before transmission[1]. This pre-allocation of network resources ensures end-to-end QoS guarantees for the traffic flow. However, due to its protocol implementation intricacies and inefficient bandwidth utilization, the IETF introduced the DiffServ model[2]. The DiffServ model distinguishes itself through its unique strategy of marking data at the network's edge nodes, which is crucial for defining the subsequent handling and processing of the transmitted data.

In 2005, the Institute of Electrical and Electronics Engineers (IEEE) established the Audio-Video Bridging (AVB) working group, aiming to develop Ethernet AVB technology[3]. AVB is a set of real-time audio and video transmission protocols based on a new Ethernet architecture. In 2012, the IEEE 802.1 task group officially renamed AVB as TSN. TSN encompasses a range of technical standards, primarily focused on clock synchronization, data stream scheduling strategies, and network and user configurations.

In 2015, the IETF established the DetNet working group, with a specific focus on achieving deterministic, worst-case bounds on delay, packet loss, and jitter by implementing deterministic transmission paths at the second-layer bridging and third-layer routing segments[4]. This allows for predictable latency in network communications.

In order to provide guidance for the development of network

This work was supported by ZTE Industry-University--Institute Cooperation Funds.

technologies and the practical planning of networks, researchers are increasingly emphasizing the use of mathematical models to analyze network performance. In earlier years, researchers employed mathematical theories such as the probability theory and queueing theory to analyze network service performance. However, as network structures became more complex and services diversified, the limitations of these theories gradually became evident. Network calculus, a theoretical framework that incorporates min-plus algebra, has been instrumental in transforming intricate nonlinear queueing problems into mathematically tractable models. Therefore, in recent years, network calculus has been proven to be an effective and versatile tool for analyzing network component performance. These components can encompass links, schedulers, shapers, or even entire networks. As a result, network calculus has found widespread application in scenarios of internet QoS[5-12].

Network calculus uses service curves to describe the service capabilities of network elements such as routers, schedulers, and links. In the min-plus algebra theory, the service curves of interconnected components can be combined through convolution, resulting in an overall service curve. Consequently, individual systems along a network path can be easily connected by convolving their service curves, thus obtaining a specified end-to-end service curve for the network. By combining the service curve with the arrival curve, network performance bounds can be determined. Different network architectures provide QoS guarantees for services within the network by employing various traffic shaping and scheduling algorithms at network nodes. Consequently, the service curves of different scheduling algorithms constitute a crucial foundation for analyzing network performance. Therefore, this paper aims to summarize the service curves for different scheduling algorithms.

The main contribution of this paper is to compile the concepts and service curves of seven typical scheduling algorithms from existing studies. The remaining content of this paper is arranged as follows: Section 2 introduces the basic concept of network calculus; Section 3 introduces different scheduling algorithms and their service curves; Section 4 calculates the delay bounds using service curves from various scheduling algorithms and conducts simulation in burst flows situations scenarios; Section 5 concludes by summarizing the contributions of the study.

2 Network Calculus

Network calculus was initially developed to analyze the performance of networks with non-probabilistic distribution traffic. The origins of network calculus can be traced back to CURZ's research on traffic characteristics and network performance boundaries[13-14], as well as the studies by PAREKH and GALLAGHER on the service curves of Generalized Processor Sharing (GPS) schedulers[15-16]. Subsequently, researchers like CRUZ and SARIOWAN advance the study of network calculus[17-23], formalizing the concept of service curves and establishing the foundational framework for network calculus[24].

Ref. [25] systematically discusses the fundamental concepts of deterministic network calculus, along with the foundation and theory of min-plus algebra, and further integrates specific analyses involving internet traffic and scheduling mechanisms. This section provides a brief introduction to the basic framework of network calculus based on the content presented in Ref. [25].

Network calculus is built upon the foundation of min-plus algebra and is used for the analysis of network performance. It involves two fundamental non-decreasing operations: min-plus convolution $\otimes$ and min-plus deconvolution $\oslash$:

$$(a \otimes b)(x) = \inf_{0 \leqslant y \leqslant x} \left[a(y) + b(x-y)\right], \tag{1}$$

$$(a \oslash b)(x) = \sup_{y \geqslant 0} \left[a(x+y) - b(y)\right]. \tag{2}$$

Both arrival curves and service curves are determined through min-plus convolution. The arrival curve $\alpha(t)$ serves as a mathematical model for the constrained arrival process $A(t)$ of a traffic flow, where $A(t)$ represents the cumulative input function, quantifying the amount of data that has arrived at the network node up to time t.

$$A(t) \leqslant A \otimes \alpha(t) = \inf_{0 \leqslant s \leqslant t} \left\{A(s) + \alpha(t-s)\right\}. \tag{3}$$

The affine arrival curve $\alpha_{r,b}(t) = rt + b$ is a common type of arrival curve. This model is frequently used to describe traffic patterns where data is transmitted at a constant rate of r after an initial burst of size b.

The service curve $\beta(t)$ describes the capacity of a node to provide services. Let the departure function be denoted as $A^*(t)$, which is the cumulative output function representing the total amount of data output from the network node up to time t. $\beta(t)$ satisfies the following relationship:

$$A^*(t) \geqslant \inf_{0 \leqslant s \leqslant t} \left\{A^*(s) + \beta(t-s)\right\} = A \otimes \beta(t). \tag{4}$$

A strict service curve $\beta(t)$ satisfies the following relationship throughout any backlog period:

$$A^*(t+\Delta t) - A^*(t) \geqslant \beta(\Delta t). \tag{5}$$

A commonly used service curve is the Latency-Rate (LR) service curve $\beta_{r,T}(t) = r[t-T]^+$, which provides a simple way to describe the worst-case behavior of various scheduling algorithms[26]. This service curve represents the service node's

guarantee of providing a service rate of r to a traffic flow, with the additional constraint that delays do not exceed T.

Network calculus encompasses several fundamental theorems that serve as the theoretical underpinnings for analyzing network performance. Here are some of the basic theorems in network calculus[27]:

1) Delay bound: The delay $D(t)$ of traffic at a network node at time t is bounded by the maximum horizontal distance between the arrival curve $\alpha(t)$ and the service curve $\beta(t)$:

$$D(t) \leqslant h(\alpha,\beta) = \sup_{s \geqslant 0}\left\{inf\left\{\tau \geqslant 0:\alpha(s) \leqslant \beta(s+\tau)\right\}\right\}. \quad (6)$$

2) Backlog bound: The backlog of traffic at a network node is bounded by the maximum vertical deviation between the arrival curve $\alpha(t)$ and the service curve $\beta(t)$:

$$B(t) \leqslant v(\alpha,\beta) = \sup_{s \geqslant 0}\left\{\alpha(s) - \beta(s)\right\}. \quad (7)$$

3) Output characterization: The deterministic arrival process of the departure process A^* can be represented by the deterministic arrival curve $\alpha^*(t)$:

$$\alpha^*(t) = \alpha \oslash \beta = \sup_{s \geqslant 0}\left\{\alpha(t+s) - \beta(s)\right\}. \quad (8)$$

4) Concatenation property: The deterministic service curve provided to data flows after the concatenation of several nodes can be represented as the convolution of these nodes' service curves:

$$\beta(t) = \beta^1(t) \otimes \beta^2(t) \cdots \otimes \beta^n(t). \quad (9)$$

5) Superposition: The arrival curve of an aggregated flow can be represented as the pointwise sum of the arrival curves of individual flows:

$$\alpha(t) = \sum_{i=1}^{n} \alpha_i(t). \quad (10)$$

6) Leftover service: This theorem addresses the number of services that remain available in a network node after some data have been served to specific flows. It is often used to calculate the remaining service capacity or service curve of a network element. For a network node that includes two flows, A_1 and A_2, with the node's service curve being $\beta(t)$, ${A^*}_1(t)$ satisfies the following equation:

$${A^*}_1(t) \geqslant A_1 \otimes \left(\beta - \alpha_2\right)^+(t). \quad (11)$$

3 Scheduling Algorithms and Service Curves

In the expansive landscape of scheduling algorithms for network communications, a multitude of strategies exist to address the diverse challenges posed by data transmission. Among this myriad of options, we focus on seven distinctive scheduling algorithms that encapsulate both classical methodologies and contemporary innovations. The selected algorithms include classical strategies such as Strict Priority (SP), Round Robin (RR), and Weighted Fair Queuing (WFQ). Furthermore, this paper delves into the advanced strategies introduced by TSN, namely Cycling Queuing and Forwarding (CQF), Time Aware Shaper (TAS), Asynchronous Traffic Shaper (ATS), and Credit Based Shaper (CBS). The specifics of these algorithms are discussed in this paper, as well as their relation to the creation of service curves.

3.1 Strict Priority

3.1.1 Scheduling Algorithm

SP is a classic and pivotal scheduling strategy known for its effectiveness in ensuring high-priority data stream transmission performance. It proves particularly suitable for applications requiring guaranteed low latency. SP strictly follows the order of queue priorities. Packets with the same priority are scheduled using the FCFS policy. Each queue's packets must wait until all packets in higher priority queues have been scheduled before they have the opportunity to be scheduled. It ensures that high-priority data receive preferential treatment within the constraints of limited resources.

3.1.2 Service Curve

In the context of preemptive SP scheduling, if data from a high-priority flow arrive at the server, they will receive an immediate service, even if data from a low-priority flow are currently being serviced. Consequently, lower-priority flows do not affect the service provided by the server to higher-priority flows. It is sufficient to analyze two types of flows: the flow with priority level i and the aggregate flow with priorities greater than i. According to the theorem of Leftover Service, the service curve for the flow with priority level i is as follows[25].

$$\beta_i(t) = \left[\beta(t) - \sum_{j=1}^{i-1} \alpha_j(t)\right]^+, \quad (12)$$

where $\beta(t)$ represents the overall service curve provided by the service node, $\alpha_i(t)$ represents the arrival curve for the data flow with priority level i, and $[x]^+$ denotes $\max(0,x)$.

In practical networks, data flows are composed of data packets and non-preemptive strategies are often employed. In a non-preemptive SP scheduling, if a low-priority data packet has started being serviced, it continues to be serviced even if higher-priority data packets arrive. In such cases, the service curve for a non-preemptive SP scheduling for a flow with priority level i is as follows[25].

$$\beta_i^{SP}(t) = \beta(t) - \sum_{j=1}^{i-1} \alpha_j(t) - \max_{j>i} l_j^u, \quad (13)$$

where l_j^u represents the maximum packet length in the queue with priority level j.

3.2 Weighted Fair Queuing

3.2.1 Scheduling Algorithm

While the SP scheduling algorithm holds a crucial position in network communications, especially for latency-sensitive applications, it exhibits noticeable limitations. In instances where high-priority data streams are overly frequent, SP may lead to prolonged neglect of low-priority data streams, impacting overall fairness. In the contemporary field of network communications, ensuring both fairness and efficiency in scheduling various priority and data stream types is paramount for QoS. To address this, researchers have introduced WFQ, a strategy that employs weight assignments and a fair queuing approach. WFQ aims to provide equitable and efficient services to different data streams by ensuring that each stream receives a fair share of resources based on its assigned weight. This approach acknowledges the importance of maintaining fairness while effectively managing the diverse priorities and types of data streams in today's network communication landscape.

WFQ allocates the bandwidth to each flow based on their queue weights. WFQ, also known as Packet General Processor Sharing (PGPS), is a variant of the idealized scheduling strategy called GPS[15-16]. GPS is a scheduling strategy that allocates a certain proportion of service guarantees to each priority queue based on weight parameters. For n queues with weight parameters $\varphi_1,\cdots,\varphi_n$, queue i is guaranteed a service proportion of $\varphi_i/\sum\varphi_j$. However, GPS is an idealized strategy that assumes each data segment can be divided into infinitely small segments to ensure proportional sharing at every point in time and data size. In practical applications, bits cannot be divided, and data packets are usually not fragmented. Researchers have introduced PGPS scheduling to approximate the ideal GPS strategy under real-world constraints.

In a WFQ system, when a packet arrives, its departure time in the corresponding GPS system is calculated. The system selects the packet with the smallest departure time for transmission. This process introduces a monotonically increasing virtual time function, denoted as $V(t)$:

$$V\left(t_{j-1}+\tau\right)=V\left(t_{j-1}\right)+\frac{\tau\cdot C}{\sum_{i\in B_j}\varphi_i},\quad \tau<t_j-t_{j-1}. \tag{14}$$

In Eq. (14), $V(0)=0$, and t_j represents the time at which the j-th event occurs in the system. These events can include either the departure or arrival of data packets. B_j represents the set of non-empty queues between the time t_{j-1} and t_j.

We can define the arrival time of the k-th data packet in queue i as a_i^k, and the length of this packet as L_i^k. Additionally, the allocation of transmission rates to each queue i in a WFQ system is represented by weight parameters φ_i. S_i^k and F_i^k represent the virtual start time and virtual finish time of the k-th data packet in queue i. If there is no special explanation, set S_i^0 to 0. When queue i is empty, $S_i^k=V(a_i^k)$, and when queue i is not empty, $S_i^k=F_i^{k-1}$.

$$S_i^k=\max\left\{F_i^{k-1},V\left(a_i^k\right)\right\},$$

$$F_i^k=S_i^k+\frac{L_i^k}{\varphi_i}. \tag{15}$$

The virtual finish time F_i^k for each queue is calculated according to Eq. (15). Then, the system selects the data packet for transmission by choosing the one with the smallest virtual finish time F_i^k among all the queues. This ensures that the packet from the queue with the earliest virtual finish time is sent next, maintaining fairness in resource allocation.

3.2.2 Service Curve

Refs. [15] and [16] introduce the concept of GPS scheduling strategy and conducted performance analysis of GPS schedulers in both single-node and multi-node scenarios. The derived expression for GPS node availability of a service is considered to be the first service curve formula in network calculus. This work laid the foundation for understanding and analyzing service curves in the context of network calculus. If data flow i experiences backlog within the time interval (s,t), the following condition holds for data flow j in all cases:

$$\phi_j\left(D_i(t)-D_i(s)\right)\geqslant\phi_i\left(D_j(t)-D_j(s)\right). \tag{16}$$

The service curve for data flow i is as follows.

$$\beta_i^{\mathrm{GPS}-1}(t)=\frac{\varphi_i}{\sum_{j=1}^{n}\varphi_j}\beta(t). \tag{17}$$

WFQ can be considered a packet-based form of GPS, which means that we can derive the service curve for data flow i within a WFQ node:

$$\beta_i^{\mathrm{WFQ}-1}(t)=[\beta_i^{\mathrm{GPS}-1}(t)-\max_{1\leqslant j<n}l_j^u]^+. \tag{18}$$

The previous conclusion assumes that all data flows are continuously backlogged, fully utilizing the allocated bandwidth. In reality, if one or more flows do not make full use of their allocated resources, the remaining service shares will be redistributed to the backlogged flows based on their weights. Ref. [26] takes into account practical scenarios and used the departure process's arrival curve to provide a more general service curve for data flow i:

$$\beta_i^{GPS-2}(t) = \max_{M \subseteq \{1,\cdots,n\}} \left\{ \frac{\varphi_i}{\sum_{j \in M} \varphi_j} \left[\beta(t) - \sum_{j \notin M} \alpha_j^*(t) \right]^+ \right\}. \quad (19)$$

The departure process's arrival curve α_j^* is determined based on the arrival curve α_j and the service curve specified in Eq. (18).

In addition, the research in Refs. [15] and [16] assumes that all flows have affine arrival curves of the form $\alpha_i(t) = r_i t + b_i$, with constant link rates, and the system is stable, ensuring that the overall average arrival rate does not exceed the link capacity. However, Ref. [28] conducted research without being constrained by these conditions and derived more general service curves:

$$\beta_i^{GPS-3}(t) = \max_{M \subseteq \{1,\cdots,n\} \setminus \{i\}} \left\{ \frac{\varphi_i}{\sum_{j \notin M} \varphi_j} \left[\beta(t) - \sum_{j \in M} \alpha_j(t) \right]^+ \right\}. \quad (20)$$

3.3 Round Robin

3.3.1 Scheduling Algorithm

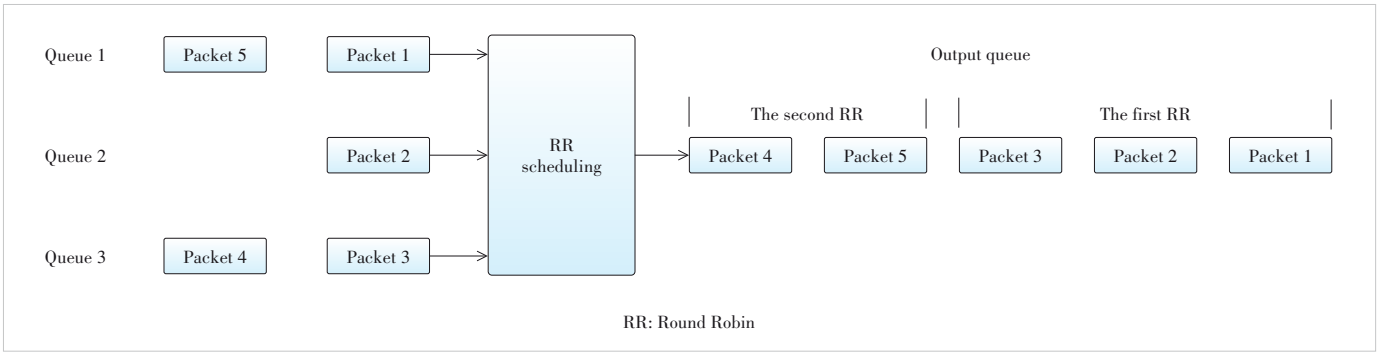
RR scheduling strategy[29] is proposed to address the limitations of SP. Similar to WFQ, RR utilizes weight assignments to enhance overall fairness within the system. RR also comes in various variants, each employing different methods of allocating transmission resources to improve overall system performance from distinct perspectives.

RR employs a polling mechanism to schedule multiple queues, and within each queue, a FCFS scheduling strategy is applied. During each polling round, the scheduler sequentially sends the first data packet from each queue, skipping over empty queues. Fig. 1 shows a scheduling scenario.

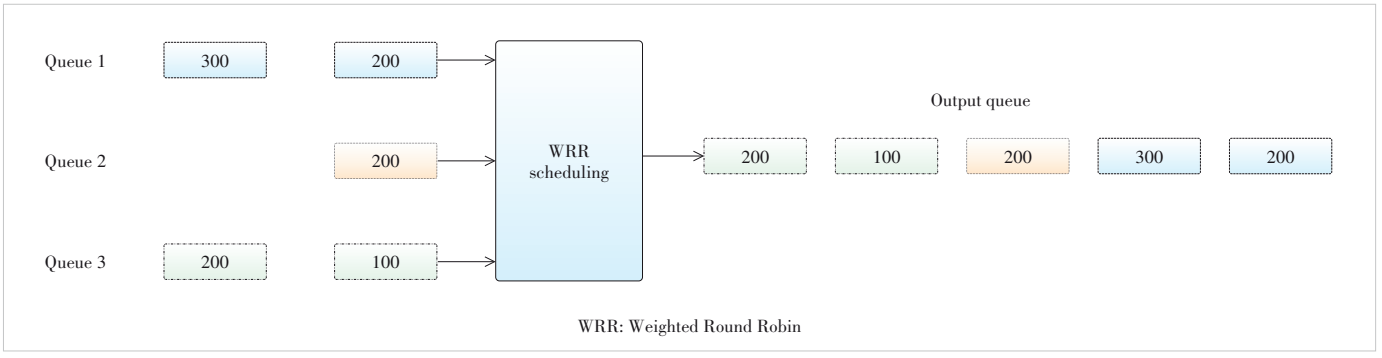
The RR algorithm provides uniform cyclic scheduling services to different queues. However, it may result in unfairness when dealing with queues with varying packet sizes. Additionally, it does not differentiate between services with different latency requirements, making it challenging to guarantee QoS for high-latency-sensitive services. As a result, various variants of RR are introduced, such as Weighted Round Robin (WRR)[30] and Deficit Round Robin (DRR)[31].

WRR allocates service proportionally to the weight of each queue during polling. In the classic WRR, during each polling round, each queue consecutively sends data packets w_i that is the weight of the queue. The standard RR scheduling can be seen as a special case of WRR, where the weight of each queue is 1. For example, in a WRR scenario with weights 2:2:1, the polling sequence would allocate services as shown in Fig. 2.

However, continuous sending of consecutive data packets w_i can lead to burstiness in data flows and potentially impact other queues. Interleaved Weighted Round-Robin (IWRR) addresses this issue by eliminating the impact through an alter-



▲Figure 1. RR scheduling



▲Figure 2. WRR scheduling

nating mechanism. In this approach (Fig. 3), each queue is assigned a counter, which is initialized based on the queue's weight. When a queue with a non-zero counter is polled, it sends one data packet and reduces its counter by 1. Once the counter reaches 0, the queue is skipped, and the scheduler moves on. A new scheduling round begins when the counters of all queues are 0.

DRR is a scheduling algorithm that operates based on packet lengths. In DRR, each queue is assigned a counter and initialized to the maximum number of bytes that can be scheduled in one round, known as Quantum. During each round of polling, when a queue is reached and if the length of a packet in the queue is less than the counter's value, the packet is sent and the counter is reduced by the length of the sent packet. This process continues until the counter's value becomes smaller than the length of the first packet in the queue, at which point the next queue is scheduled. After each polling round, the counter is reset to its maximum value and a new scheduling round begins.

For example, when all queues have a Quantum of 200, the DRR scheduling mechanism functions as illustrated in Fig. 4.

3.3.2 Service Curve

Researchers have extensively studied the service curves of RR and its variants.

For WRR, it is evident that the packet length has a significant impact on the performance of the WRR algorithm. The researchers in Ref. [32] model the packet length sequence and provide three different service curve models with varying levels of precision and complexity:

$$\beta_i^{WRR-1}(t) = \frac{q_i}{q_i + Q_i}\left[\beta(t) - Q_i\right]^+, \tag{21}$$

$$\beta_i^{WRR-2}(t) = \lambda_1 \otimes \upsilon_{q_i, q_i+Q_i}\left(\left[\beta(t) - Q_i\right]^+\right), \tag{22}$$

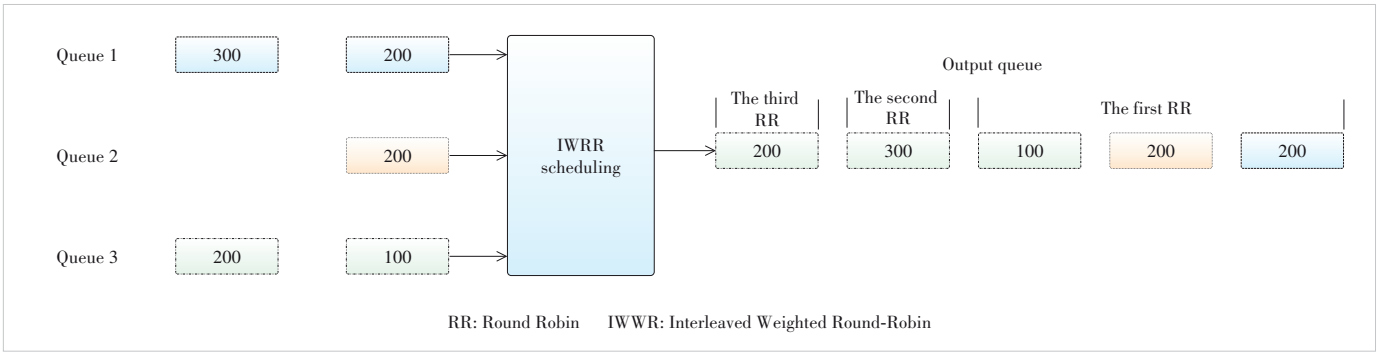
$$\beta_i^{WRR-3}(t) = f_i^{-1}\left(\beta(t)\right), \tag{23}$$

where $Q_i = \sum_{j \neq i} w_j l_j^u$ represents the maximum amount of data that all other queues can receive in one round of polling and $q_i = w_i l_i^l$ represents the minimum amount of data that queue i can receive in one round of polling (l_i^u and l_i^l represent the maximum and minimum packet lengths in queue i and w_i is the WRR weight of the queue).

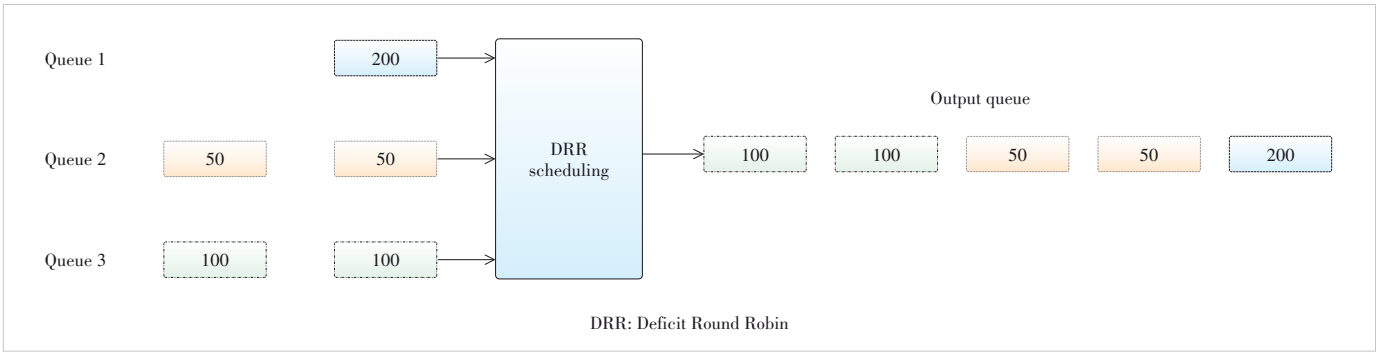
Functions $\upsilon_{a,b}(t)$, $\lambda_k(t)$ and $f_i(x)$ can be expressed as:

$$\upsilon_{a,b}(t) = a\left\lceil \frac{t}{b} \right\rceil, \ t > 0, \tag{24}$$

$$\lambda_k(t) = kt, \ t > 0, \tag{25}$$



▲Figure 3. IWWR scheduling



▲Figure 4. DRR scheduling

$$f_i(x) = x + \sum_{j \neq i} L_j^u \left(w_j \left\lfloor 1 + \frac{g(x)}{w_i} \right\rfloor \right), \tag{26}$$

$$g = \sup\left\{ x | L_i^l(x) \leqslant y \right\}, \tag{27}$$

where L_i^u and L_i^l represent the upper and lower bounds of the cumulative packet length sequence of data stream i, respectively. The data packet length sequence L_i satisfies the following condition:

$$\forall j, n \in N, L_i^l(n) \leqslant \sum_{j}^{j+n-1} L_i(j) \leqslant L_i^u(n). \tag{28}$$

Among the three service curves mentioned above, Eq. (23) is the most accurate, as it utilizes the data packet curve. However, it is computationally more complex.

If the maximum packet length L_i^u and minimum packet length L_i^l are known, we can get $L_i^u(n) = nl_i^u(n)$ and $L_i^l(n) = nl_i^l$. In this case, Function $f_i(x)$ can be simplified as:

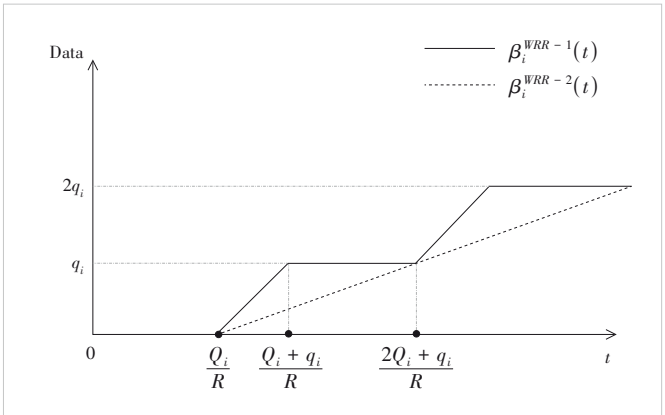
$$f_i(x) = x + Q_i \left\lfloor \frac{x}{w_i} \right\rfloor + Q_i. \tag{29}$$

In this case, the inverse function takes the form of $\lambda_1 \otimes \upsilon_{q_i, q_i + Q_i}$, which results in the service curve being transformed into the service curve in Eq. (22).

Eq. (22) obtains the details of the polling using a pseudo-inverse method, taking into account the total bandwidth used by the packets currently being serviced. On the other hand, Eq. (21) can be viewed as the linearized result of Eq. (22) and their service curve is depicted in Fig. 5.

For IWRR, Ref. [33] employs a method similar to Eq. (22) to obtain the service curve for IWRR using a pseudo-inverse:

$$\beta_i^{IWRR}(t) = \lambda_1 \otimes \sum_{k=1}^{w_i - 1} \upsilon_{l_i^l, L_{tot}} \left(\left[\beta(t) - \psi_i^{IWRR}\left(k l_i^l \right) \right]^+ \right), \tag{30}$$



▲**Figure 5. Service curve for $\beta_i^{WRR-1}(t)$ and $\beta_i^{WRR-2}(t)$**

$$\psi_i^{IWRR}(x) = x + \sum_{j \neq i} \theta_{i,j}^{IWRR} \left(\left\lfloor \frac{x}{l_i^l} \right\rfloor \right) l_j^u, \tag{31}$$

$$\theta_{i,j}^{IWRR}(x) = \left\lfloor \frac{x}{w_i} \right\rfloor w_j + \left[w_j - w_i \right]^+ + \min\left(x \bmod w_i + 1, w_j \right). \tag{32}$$

For DRR, Ref. [31][34][35] study the worst-case performance under strict assumptions, mainly assuming that the server has a constant service rate. On the other hand, Ref. [36] uses network calculus to analyze the delay of DRR in a more general scenario, encompassing the results from Refs. [31], [34] and [35]. It derives the DRR service curve as:

$$\beta_i^{DRR}(t) = \left[\frac{Q_i}{F} \beta(t) - \frac{Q_i\left(L - l_i^u\right) + \left(F - Q_i\right)\left(Q_i + l_i^u\right)}{F} \right]^+, \tag{33}$$

where $F = \sum_{i=1}^{n} Q_i$, representing the total maximum number of bytes processed in one round, and $L = \sum_{i=1}^{n} l_i^u$, representing the sum of the upper bounds of packet sizes from all queues. Additionally, considering that packet sizes are discrete and multiples of a base unit ε (e.g., 1 byte), we define $l_i^{u-\varepsilon} = l_i^u - \varepsilon$ and $L = \sum_{i=1}^{n} l_i^{u-\varepsilon}$. The service curve is defined as follows:

$$\beta_i^{DRR-\varepsilon-1}(t) = \left[\frac{Q_i}{F} \beta(t) - \frac{Q_i\left(L^\varepsilon - l_i^{u-\varepsilon}\right) + \left(F - Q_i\right)\left(Q_i + l_i^{u-\varepsilon}\right)}{F} \right]^+. \tag{34}$$

To some extent, the polling process can be considered as a simulation of GPS. Therefore, the analysis of RR can also leverage research on GPS. The researchers in Ref. [37] introduce a more comprehensive concept, the bandwidth sharing policy, to unify GPS and RR. They introduce a new method to derive the service curves for bandwidth-sharing scheduling policies and improve the performance boundaries by exploiting the characteristics of cross traffic. They prove that for a variable-capacity network node with service curve $\beta(t)$ and bandwidth-sharing parameters $\varphi_j, 1 \leqslant j \leqslant n$, if $\beta(t)$ is a convex function and all data flow arrival curves $\alpha_j(t)$ are concave functions, there exists a non-negative integer set H_M, $M \subseteq \{1, \cdots, n\} / \{i\}$, and then the service curve for data flow i can be expressed as:

$$\beta_i^{BS}(t) = \sup_{M} \left\{ \frac{\varphi_i}{\sum_{j \notin M} \varphi_j} \left[\beta(t) - \sum_{i \in M} \alpha_i(t) - H_M \right]^+ \right\}. \tag{35}$$

Ref. [37] introduces an inductive process to compute H_M and applies the conclusion to GPS and DRR. For GPS, in the

particular case where H_M=0, they obtain the same result as in Ref. [28]. For DRR, the service curve can be rewritten as:

$$\beta_i^{BS-DRR}(t) = \sup_M \left\{ \frac{Q_i}{\sum_{j \notin M} Q_j} \left[\beta(t) - \sum_{i \in M} \alpha_i(t) - H_M \right]^+ \right\}. \quad (36)$$

Furthermore, Ref. [38] makes further advancements using the pseudo-inverse and output arrival curves from the research in Refs. [36] and [37]. The pseudo-inverse is initially used to gain more insights into DRR, improving the service curve for DRR and resulting in the new service curve in the absence of arrival constraints.

$$\beta_i^{DRR-\varepsilon-2}(t) = Y_i(\beta(t)), \quad (37)$$

$$Y_i(x) = \lambda \otimes v_{Q_i, Q_{tot}}\left(\left[x - \psi_i\left(Q_i - l_i^{u-\varepsilon}\right) \right]^+ \right) + \min\left(\left[x - \sum_{j \neq i} \left(Q_j + l_j^{u-\varepsilon}\right) \right]^+, \left(Q_i - l_i^{u-\varepsilon}\right) \right), \quad (38)$$

$$Q_{tot} = \sum_{j=1}^{n} Q_j, \quad (39)$$

$$\psi_i^{DRR}(x) = x + \sum_{j \neq i} \theta_{i,j}^{DRR}(x), \quad (40)$$

$$\theta_{i,j}^{DRR}(x) = \left\lfloor \frac{x + l_i^{u-\varepsilon}}{Q_i} \right\rfloor Q_j + \left(Q_i + l_j^{u-\varepsilon}\right). \quad (41)$$

Moreover, Ref. [38] introduces an iterative method to improve the service curve by taking into account arrival curve constraints of interfering flows (with concave arrival curves). This method is applicable to any available strict service curves for DRR.

$$\beta_i^{new}(t) = \max\left(\beta_i^{old}(t), \max_{M \subseteq \{1,\cdots,n\} \setminus \{i\}} Y_i^M \left(\left[\beta(t) - \sum_{j \in \{1,\cdots,n\} \setminus \{i\} \setminus M} \alpha_j \oslash \beta_i^{old}(t) \right]^+ \right)\right), \quad (42)$$

$$Y_i^M(x) = \lambda \otimes v_{Q_i, Q_{tot}^{M,i}}\left(\left[x - \psi_i^M\left(Q_i - l_i^{u-\varepsilon}\right) \right]^+ \right) + \min\left(\left[x - \sum_{j \in M} \left(Q_j + l_j^{u-\varepsilon}\right) \right]^+, \left(Q_i - l_i^{u-\varepsilon}\right) \right), \quad (43)$$

$$Q_{tot}^{M,i} = Q_i + \sum_{j \in M} Q_j, \quad (44)$$

$$\psi_i^M(x) = x + \sum_{j \in M} \theta_{i,j}(x). \quad (45)$$

On the other hand, inspired by Ref. [37], the researchers in Ref. [39] introduce a new service curve for WRR that demonstrates improved performance bounds when dealing with cross-traffic and arrival constraints. Assuming a set of flows $N = \{1,\cdots,n\}$ and each flow i is constrained by concave arrival curves $\alpha_i(t)$, the following relationship holds:

$$\beta_i^{BS-WRR}(t) = \sup_{i \in M \subset N} \left\{ \frac{q_i}{q_i + Q_i^M} \left[\beta(t) - \sum_{j \notin M} \alpha_j(t) - Q_i^M \right]^+ \right\}, \quad (46)$$

$$Q_i^M = \sum_{j \in M \setminus \{i\}} w_j l_j^u. \quad (47)$$

3.4 Cycling Queuing and Forwarding

3.4.1 Scheduling Algorithm

The emergence of TSN has brought about enhanced latency determinism and QoS in network communications. The CQF[40] scheduling algorithm, as a crucial component in the field of TSN, plays a vital role in efficiently forwarding data while ensuring fairness. CQF scheduling, through its cycling queuing and precise forwarding mechanism, effectively enhances the network's ability to control latency, reduce data transmission jitter, and simultaneously improve overall network throughput. This algorithm not only safeguards critical data but also contributes to the overall efficiency of the network.

CQF utilizes dual queues in conjunction with gating control principles and service scheduling strategies. It requires precise clock synchronization support. In the CQF mechanism, gate structures are applied exclusively at the output ports of the data buffer queues. When the gate is open, data from the queue is allowed to be forwarded to the next node. Conversely, when the gate is closed, incoming data is buffered within the queue, awaiting transmission.

3.4.2 Service Curve

Considering CQF alternates transmission using two identical queues, Ref. [41] divides the service model into odd and even queues and models the queues as greedy shapers. It constructs the shaping curves based on the service model of time division multiple access (TDMA) from Ref. [42]. Finally, it derives the service curves for the two queues in CQF.

$$\beta^{CQF-1}(t) = \sigma_{odd}(t) = C \cdot \min\left(\left\lfloor \frac{t}{2T_Q} \right\rfloor \cdot T_Q, t - \left\lfloor \frac{t}{2T_Q} \right\rfloor \cdot T_Q \right), \quad (48)$$

$$\beta^{CQF-0}(t) = \sigma_{even}(t) = C \cdot \max\left(\left\lfloor \frac{t}{2T_Q} \right\rfloor \cdot T_Q, t - \left\lfloor \frac{t}{2T_Q} \right\rfloor \cdot T_Q \right), \quad (49)$$

where $\beta^{CQF-1}(t)$ is the service curve for the odd queue, β^{CQF-0} is the service curve for the even queue, T_Q is the alternating queue period, and C is the port forwarding rate.

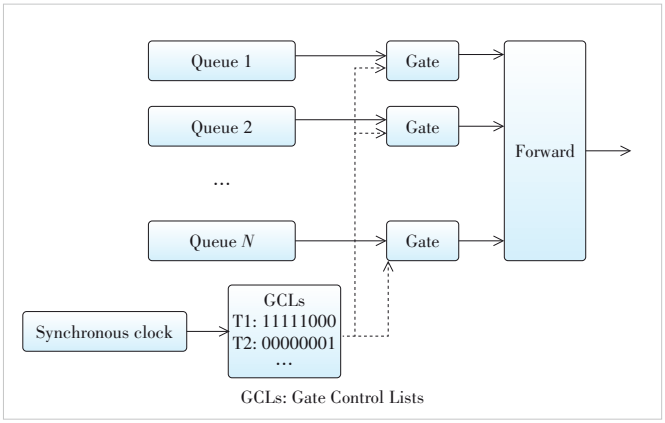
3.5 Time Aware Shaper

3.5.1 Scheduling Algorithm

In industrial IoT and similar contexts, there is a need for stringent requirements in terms of latency and jitter for certain types of data. Exceeding specified thresholds for either latency or jitter can potentially lead to severe consequences. Moreover, such data are often transmitted periodically. To meet the performance demands of these scenarios, TSN introduces scheduled traffic (ST)[43] to support low latency and low jitter applications. Additionally, the TSN framework includes TAS, which ensures that ST flows receive the necessary latency guarantees. TAS relies on highly precise clock synchronization to implement a gate-based scheduling mechanism. TAS periodically scans predefined Gate Control Lists (GCLs) to control the opening and closing of gates associated with different queues. When a gate is open, data frames in the corresponding queue can be transmitted, and when it is closed, they await their turn for transmission. CQF can be perceived as a solution built upon TAS. Fig. 6 shows the structure of TAS.

To ensure that low-priority data frames do not impact the transmission of high-priority data frames, TAS introduces the concept of the guard band (GB). Before the window for high-priority data frames opens, a segment of time equivalent to the GB duration is reserved. During this GB period, new data frames cannot begin transmission. The typical length of the GB is set to the maximum data frame transmission time within the communication network.

To address the issue of resource wastage caused by GB, TSN introduces the frame preemption (FP) mechanism[44]. This mechanism classifies frames based on their priority. The transmission of low-priority frames will be interrupted when high-priority frames arrive, and only resumes after the high-priority frames have been completely transmitted. When a high-priority frame arrives, the system checks if the remaining segment of the low-priority frame being transmitted satisfies the slicing conditions. If it does, the low-priority frame's transmission is paused, and a 4-byte Message Checksum Redundancy Check (MCRC) is added to the already transmitted portion, effectively functioning as a checksum. This enables the assembly of the transmitted frame segments into a complete data frame. The transmission of the high-priority frame begins when a frame interval is available. Once the high-priority frame's transmission is completed, the remaining part of the low-priority frame is supplemented with a preamble code containing assembly information associated with the previously transmitted portion. Subsequently, the low-priority frame's transmission continues.



▲Figure 6. Structure of Time Aware Shaper (TAS)

3.5.2 Service Curve

Through the careful construction of GCLs, TAS can periodically reserve transmission resources for different traffic classes, thereby providing reliable QoS guarantees. It can also offer a completely deterministic transmission mode for individual flows. In this fully deterministic mode, there is no need for network calculus analysis, but it has limited application scenarios. Therefore, as discussed in Ref. [45], the analysis of the delay bounds for ST flows focuses on the most general case without specifying the construction method of the GCL.

TAS shares some similarities with TDMA in the sense that they both allocate transmission resources to different services based on a time scale. However, TDMA allows different services to transmit in non-overlapping time slots, which eliminates conflicts between them. In contrast, in the TAS mechanism, when multiple gates are open simultaneously, lower-priority data frames must wait for the completion of the transmission of higher-priority data frames before they can start the transmission. Additionally, without utilizing the FP mechanism, they may also need to wait for even lower-priority data frames to complete their transmission.

Within a GCL period T_{GCL}, the data frames of different priority levels have varying periods and the regions of overlapping gate opening times for different priority queues differ. This results in varying lengths L of each transmission window during each GCL cycle. Let N represent the number of priority queues, S_i denote the interval between the reference position (the starting time of the GCLs period) and the first transmission window for priority queue i, and o_i^j represents the relative offset from the j-th transmission window of queue i to the first transmission window. N_i is the number of transmission windows within one GCL cycle for queue i. In the study of Ref. [44], these transmission window parameters are used to calculate length L_i^j of the j-th transmission window for queue i. Combining this with service model $\beta_{T,L}$ based on TDMA[42], the authors obtain the service curve provided by TAS nodes for data flows with priority level M in the non-preemptive mode.

$$\beta_i^{TAS}(t) = \sum_{j=0}^{N_i-1} \beta_{T_{GCL},L_i^j}\left(t + T_{GCL} - L_i^j - S_i - o_i^j\right), \tag{50}$$

$$\beta_{T,L} = C \cdot \max\left(\left\lfloor \frac{t}{T} \right\rfloor \cdot L, t - \left\lceil \frac{t}{T} \right\rceil \cdot (T - L)\right). \tag{51}$$

For TAS and other scheduling algorithms based on precise global synchronized clocks, the service curves discussed in this paper are related to the choice of reference time points. In Eq. (41), S_i represents the time waited from the reference time point to the time when the gate of queue i is opened, and it has a significant impact on the calculated delay results. In the case of CQF in Section 3.4, the different reference points form the distinction between odd and even queues, but in reality, both queues are equivalent.

3.6 Credit Based Shaper

3.6.1 Scheduling Algorithm

To efficiently transmit audio and video data in a local area network, the IEEE 802.1 formed the AVB task group in 2005, which introduced a series of standards. Among these, the 802.1QAV[46] presents the concept of CBS. It classifies data streams into stream reservation (SR) flows and BE flows. SR flows indeed have higher latency requirements and are granted higher priority compared to BE flows. Furthermore, different SR flows can have distinct priority levels among themselves.

For SR flows, CBS utilizes credits (*credit*) to indicate whether their data can be transmitted. When *credit* exceeds 0, the corresponding data category can commence transmission. If the data for the corresponding category is either awaiting transmission or its corresponding queue is empty while the *credit* is less than 0, the *credit* increases at a rate according to *idleSlope* and decreases during transmission at a rate defined by *sendSlope*. Typically, *sendSlope* = *idleSlope* − R, where R represents the node's forwarding rate. The bounds for credit in the presence of two different SR flows are as follows[47].

$$sendSlope_A * \frac{l_A^u}{R} \leqslant credit_A \leqslant idleSlope_A * \frac{l_n^u}{R}, \tag{52}$$

$$sendSlope_B * \frac{l_B^u}{R} \leqslant credit_B \leqslant idleSlope_B * \left(\frac{l_{BE}^u + l_A^u}{R} - l_n^u * \frac{idleSlope_A}{sendSlope_A * R}\right). \tag{53}$$

Fig. 7 shows the operation of CBS, considering two categories of SR flows, A and B.

In Fig. 7, at time t_1, BE frames arrive. After time t_2, Class-A and Class-B data frames arrive. Subsequently, Class-A and Class-B data frames continuously arrive. During the time interval from t_2 to t_3, the BE frame is transmitted, and Class A and Class B are waiting. Their credits increase at rates of $idleSlope_A$ and $idleSlope_B$, respectively. When BE frame transmission finishes, both Class-A and Class-B credits are greater than 0. Class A, having a higher priority, begins its transmission. Its credit decreases at the rate of $sendSlope_A$. Class-B credit is still increasing at the rate of $idleSlope_B$. After Class-A transmission is completed, Class B follows. At this point, no Class-A frames are waiting in the queue, but its credit is less than 0. The credit increases at the rate of $idleSlope_A$ until it reaches 0. Once Class-B transmission is finished, the credit becomes greater than 0, but there are no Class-B frames left in the queue, so the credit is reset to 0.

3.6.2 Service Curve

In the context of CBS service curves, extensive research has been conducted by AZUA[47], ZHAO[48–50] and MOHAMMADPOUR[10] among others.

The study in Ref. [47] employs network calculus to model AVB networks and derives the LR service curve for CBS nodes serving Class-A and Class-B data flows:

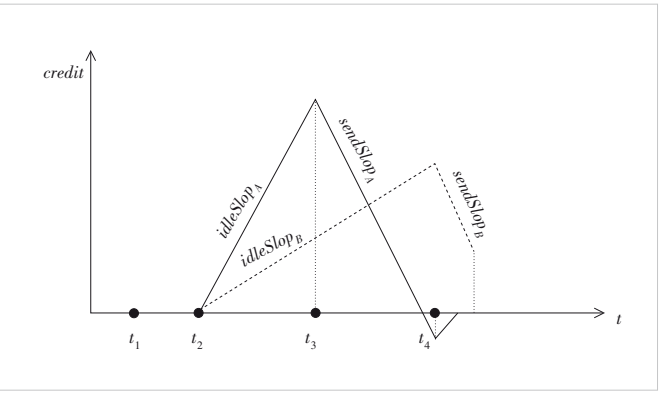
$$\beta_X^{CBS}(t) = R_X\left[t - T_X\right]^+, X = A \text{ or } B, \tag{54}$$

$$R_A = \frac{idSl_A * R}{idSl_A - sdSl_A}, T_A = \frac{l_n^u}{R} - l_A^u \frac{sdSl_A}{idSl_A * R}, \tag{55}$$

$$R_B = \frac{idSl_B * R}{idSl_B - sdSl_B}, T_B = \frac{l_B^u + l_n^u}{R} - \frac{l_n^u}{R}\frac{idSl_A}{sdSl_A} - \frac{l_B^u}{R}\frac{sdSl_B}{idSl_B}, \tag{56}$$

where $idSl_X$ and $sdSl_X$ respectively represent $idleSlope_X$ and $sendSlope_X$, and $X = A$ *or* B.

Recognizing that the two SR classes may not meet the diverse requirements of various traffic types in AVB networks, Ref. [48] establishes two types of worst-case delay models and proposes an approach to calculate the worst-case queuing de-



▲Figure 7. Changes in credit during Credit Based Shaper (CBS) operation

lay for additional SR data streams. This results in the LR service curve provided by CBS nodes for the additional SR data streams. Rate R_N corresponds to $idSl_N$ and Delay T_N represents the queuing delay experienced by the first frame of additional SR data in the worst-case scenario, which is calculated based on the worst-case queuing delay in Ref. [47].

With the rapid development of communication networks, there is an increasing diversity in the types of services and traffic in the network. To provide service guarantees for a wide range of applications, related standards have continuously expanded, defining various traffic categories with different latency requirements. Researchers have also begun to apply network calculus to analyze SR flows in situations where multiple traffic categories are mixed.

In the study of Ref. [10], control data traffic (CDT) with higher priority than Class-A and Class-B flows is considered, which extends the conclusions from Ref. [46]. If the CDT flow has an affine arrival curve $\alpha_{CDT}(t) = r_{CDT}t + b_{CDT}$, a new service curve can be derived with the following parameters:

$$R_A^{CDT} = \frac{idSl_A\left(R - r_{CDT}\right)}{idSl_A - sdSl_A}, T_A^{CDT} = \frac{1}{R - r_{CDT}}\left(l_n^u + b_{CDT} + \frac{l^u * r_{CDT}}{R}\right), \tag{57}$$

$$R_B^{CDT} = \frac{idSl_A\left(R - r_{CDT}\right)}{idSl_A - sdSl_A}, T_B^{CDT} = \frac{1}{R - r_{CDT}}\left(l_E^u + l_A^u + b_{CDT} - \frac{l_n^u * idSl_A}{sdSl_A} + \frac{l^u * r_{CDT}}{R}\right). \tag{58}$$

On the other hand, Refs. [49] and [50] derive the latency bounds for SR flows under the presence of ST flows and provide service curves for SR flows in both non-preemptive and preemptive modes.

The researchers in Ref. [49] establish the arrival curve for aggregated ST flows based on the ST window, allowing the derivation of service curves for Class-A and Class-B data flows at network nodes. Similar to Ref. [45], an ST window is defined as the time interval when the gate of ST queues opens and closes within a GCLs period T_{GCL}. In each period, there are N windows, and the length of the i-th window is denoted as L_i. The relative offset between the i-th and j-th windows, which represents the time gap between the opening times, is denoted as o_i^j. Based on this, the aggregated arrival curve for ST flows is referenced to the i-th ST window and is given by the following equation:

$$\alpha_{ST,i}(t) = \sum_{j=i}^{i+N-1} L_j R\left\lfloor \frac{t - o_i^j}{p_{GCL}} \right\rfloor. \tag{59}$$

If the impact of GB is considered, letting $L_{GB,i}$ represent the length of GB for the i-th window, the aggregated ST flow arrival curve can be expressed as:

$$\alpha_{GB+ST,i} = \sum_{j=i}^{i+N-1} R\cdot(L_j + L_{GB,j})\cdot\left\lceil \frac{t - o_i^j + L_{GB,j} - L_{GB,i}}{T_{GCL}} \right\rceil. \tag{60}$$

In the preemptive mode, when SR frame transmissions are interrupted and need to be supplemented with a preamble code to associate it with the transmitted portion, the arrival curve for the preamble code can be defined as follows if the time overhead of the preamble code is L_{OH}.

$$\alpha_{OH,i}(t) = \sum_{j=i}^{i+N-1} R * L_{OH}\left\lceil \frac{t - o_i^j - L_j}{T_{GCL}} \right\rceil. \tag{61}$$

It is assumed in Ref. [49] that when an ST queue's gate is open, gates for other queues are closed. The service curve provided by the network node for SR flows is derived with reference to the i-th window as follows:

$$\beta_{X,i}^{CBS-ST-1}(t) = \frac{idSl_X * R}{idSl_X - sdSl_X}\left[\sup_{0 \le u \le t}\left\{u - T_{X,i}^{\mathrm{mod}}(u)\right\}\right]^+, \tag{62}$$

where $mod \in \{np, p\}$ represents non-preemptive and preemptive modes, and $X \in \{A, B\}$. $T_{X,i}^{\mathrm{mod}}$ can be expressed as:

$$T_{X,i}^{np}(u) = \frac{\alpha_{GB+ST,i}(u)}{R} + \frac{credit_X^{\max}}{idSl_X}, \tag{63}$$

$$T_{X,i}^{p}(u) = \frac{\alpha_{ST,i}(u)}{R} + \frac{\alpha_{OH,i}(u)}{R}\frac{\left(idSl_X - sdSl_X\right)}{idSl_X} + \frac{credit_X^{\max}}{idSl_X}. \tag{64}$$

However, in Ref. [49], it is assumed that credits remain frozen during the GB period, which contradicts the TSN standard. Additionally, the study in Ref. [50] supports only two SR classes. Ref. [50] extends the conclusions in Ref. [49] to multiple SR classes and the case where credits are not frozen during the GB period. The bound for the credits is provided and the service curves for SR flows are derived as:

$$credit_X^{\min} = sdSl_X * \frac{l_X^u}{R}, \tag{65}$$

$$redit_X^{NF-\max} = idSl_X\frac{\sum_{j=1}^{X-1} credit_j^{\min} - l_{>X}^u - \sigma_{GB}}{\rho_{GB} + \sum_{j=1}^{X-1} idSl_j - R}, \tag{66}$$

$$credit_X^{F-\max} = idSl_X\frac{\sum_{j=1}^{X-1} credit_j^{\min} - l_{>X}^u}{\sum_{j=1}^{X-1} idSl_j - R}, \tag{67}$$

$$\beta_X^{CBS-ST-2}(t)=\frac{idSl_X*R}{idSl_X-sdSl_X}\left[t-\frac{\alpha_X^I(t)}{R}-\frac{credit_X^{I-\max}}{idSl_X}\right]^+, \quad (68)$$

where $I\in\{F,NF\}$ represents whether credits freeze (F) or do not freeze (NF) during the GB period, $X\in[1,M]$ represents the maximum number of SR flow classes, which can be up to 6, $credit_X^{I-max}$ refers to the upper bound of the credit value for an SR flow of class X in the context of mode I, and the expression $\alpha_X^I(t)$ refers to the arrival curve for an SR flow of class X:

$$\alpha_X^{NF}(t)=\max_{1\leqslant i\leqslant N}\left\{\alpha_{ST,i}(t)\right\}, \quad (69)$$

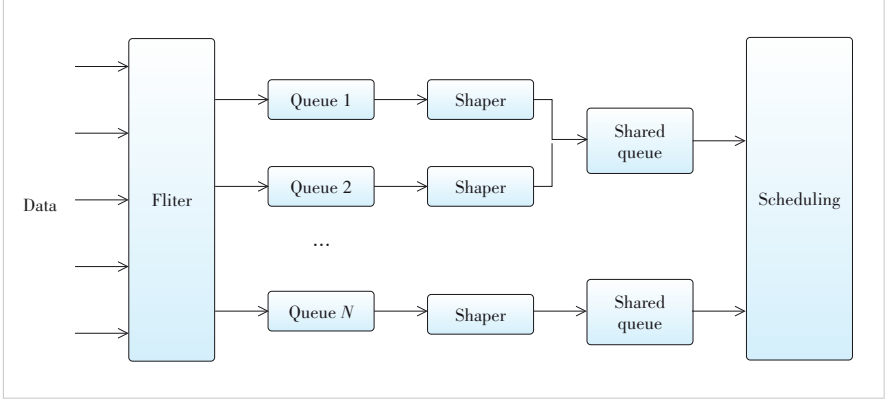
$$\alpha_X^{F}(t)=\max_{1\leqslant i\leqslant N}\left\{\alpha_{GB+ST,i}(t)\right\}. \quad (70)$$

3.7 Asynchronous Traffic Shaper

3.7.1 Scheduling Algorithm

In addition to TAS and CBS, the TSN working group has also developed the ATS[51] mechanism based on the Urgency-Based Scheduler (UBS)[8]. ATS is designed to reshape asynchronous traffic flows in TSN networks, offering an alternative solution to predictable and real-time communications. ATS operates without the need for global clock synchronization, making it suitable for scenarios where asynchronous traffic needs to be regulated to ensure reliable communications. ATS reshapes traffic at each network hop, reducing burstiness and helping meet timing constraints for various applications. Fig. 8 shows the structure and component parts of ATS.

In ATS, the flow filter first filters out data frames that exceed the size limit. It then determines the internal priority of data streams based on the priority field provided by the virtual local area network (VLAN) tag and parameters associated with flow identification. Data streams with the same priority are grouped into a shared queue. This approach is implemented to effectively manage data streams with similar QoS requirements. To mitigate burstiness in traffic, data streams are required to pass through a shaper before entering shared queues. The shaper is a kind of minimal interleaved regulator. Finally, the ATS node schedules and forwards data from all shared queues.



▲Figure 8. Structure of Asynchronous Traffic Shaper (ATS)

3.7.2 Service Curve

Inspired by Ref. [10], the researchers in Ref. [7] analyze ATS using network calculus and divide it into two parts: shaping queues and shared queues. Shared queues are scheduled using SP scheduling. For a shared queue Q_i with priority i, its service curve is as follows:

$$\beta_{Q_i}^{ATS}(t)=R\left[t-\frac{\sum_{j=1}^{i-1}\alpha_{Q_j}(t)+\max_{j>i}\left\{l_{Q_j}^u,l_{BE}^u\right\}}{R}\right]^+, \quad (71)$$

where $\alpha_{Q_i}(t)$ represents the arrival curve for the shared queue, which is determined by the output arrival curve $\alpha_q^*(t)$ of shaping queue q. $\alpha_q^*(t)$ is the sum of the output arrival curves for the various flows f in the aggregated flow of queue q. ATS uses token bucket shaping, where the token bucket's burst size and committed transmission rate are denoted as b_f and r_f, respectively. This can be expressed as:

$$\alpha_q^*(t)=\sum_{f\in q}r_f\cdot t+b_f, \quad (72)$$

$$\alpha_{Q_i}(t)=\sum_{q\in Q_i}\alpha_q^*(t). \quad (73)$$

According to the theorem of delay bound, the upper bound of delay $D_{Q_i}^{ATS}$ can be derived from $\alpha_{Q_i}(t)$ and $\beta_{Q_i}^{ATS}(t)$.

For the shaping queue, the ATS shaper at the back of the FCFS system does not increase the overall system's upper bound on delay. This is a key consideration in understanding how the shaping queue impacts the network. Hence, the delay in the shaping queue, denoted as d_q^{ATS} and the delay in the previous node's shared queue, denoted as $d_{Q_i^-}^{ATS}$, satisfy the relationship $d_q^{ATS}+d_{Q_i^-}^{ATS}\leqslant D_{Q_i^-}^{ATS}$. Additionally, the minimum delay that queue q can experience in Q_i^- is l_q^{min}/C. Therefore, the delay bound for the shaping queue is given by $D_q^{ATS}=D_{Q_i^-}^{ATS}-l_q^{min}/C$. Then Ref. [7] provides the service curve for the shaping queue based on the discussion above:

$$\beta_q^{ATS}(t)=\begin{cases}0, t\leqslant D_q^{ATS}\\ +\infty, t>D_q^{ATS}\end{cases}. \quad (74)$$

The arrival curve $\alpha_q^{ATS}(t)$ for queue q is determined by the output arrival curve from Q_i^-, and not all flows from Q_i^- will be forwarded to q. Therefore, for queue q, the output arrival curve from Q_i^- is given by:

$$\alpha_{Q_i^-}^*(t) = \frac{\sum_{f \in [Q_i^-, q]} \left(r_f \cdot t + b_f\right)}{\delta_D^{Q_i^-}(t)}, \tag{75}$$

where $\delta_D^{Q_i^-}$ is 0 for $t \leqslant D_q^{ATS}$ and $+\infty$ for $t > D_{Q_i^-}^{ATS}$. Taking into account the physical link constraints that are governed by the link-shaping curve $\delta(t) = Ct$, we have:

$$\alpha_q^{ATS}(t) = \min\left\{\alpha_{Q_i^-}^*(t), \delta(t) + l_{Q_i^-}^{\max}\right\}. \tag{76}$$

By using Eqs. (70 – 75), the delay bounds for all nodes along each flow path can be obtained. Summing up the delays of all nodes can get the end-to-end delay bound.

4 Delay Bound

In this section, we will incorporate specific scenarios and leverage the service curves and the theorem of delay bound discussed earlier to provide theoretical latency bounds for various traffic flows at a single node under different scheduling algorithms. Additionally, the open-source simulation tool NS3 is utilized to construct a simulation platform that allows the configuration of various scheduling algorithms, and the maximum latency experienced by all flows through plenty of simulations is recorded for comparison. This section will conduct a comparative analysis between the simulation results and the theoretical outcomes.

The paper considers a node with a forwarding rate R of 10 Mbit/s, and the service curve for this node is as follows:

$$\beta(t) = Rt. \tag{77}$$

Furthermore, we assume several flows of different priorities or categories passing through the current network node. All flows are periodic burst traffic, represented as (τ_i, σ_i). Here, τ_i represents the period, and σ_i represents the burst size, which is the total size of burst packets within one period. All packets are 64 bytes. The arrival curve $\alpha_i(t)$ can be expressed using the following equation:

$$\alpha_i(t) = \frac{\sigma_i}{\tau_i} t + \sigma_i. \tag{78}$$

According to the theorem of delay bound, the upper bound of the latency for different business flows passing through this node can be expressed as:

$$d_i = h(\alpha_i, \beta_i) = \sup_{s \geqslant 0}\left\{\inf\left\{\tau \geqslant 0 : \alpha_i(s) \leqslant \alpha_i(s + \tau)\right\}\right\}. \tag{79}$$

For SP scheduling, considering three different priority periodic flows, the configurations and latency bounds for different flows are shown in Table 1.

For WFQ scheduling, we consider three flows with different weight configurations. The configurations for different traffic flows are presented in Table 2. The delay bounds for different data flows using different service curves are outlined in Table 3, where β_i^{WFQ-i}, i = 1,2,3, represents the WFQ service curves derived from Eqs. (18), (20) and (21).

Although the simulation results are bounded by all theoretical results, β_i^{WFQ-1} assumes that other flows fully utilize the allocated resources, which is not consistent with the scenario presented in this paper. Consequently, the obtained latency bounds are relatively loose. β_i^{WFQ-2} and β_i^{WFQ-3} consider the possibility that each flow may not fully utilize the allocated resources, rather than allocating bandwidth directly according to weights. This results in more stringent latency bounds. For RR scheduling, we consider three flows and allocate weights based on their data arrival rates. We set the base unit ε of DRR as 64 bytes. The configurations and delay bounds for different traffic flows are presented in Tables 4 and 5.

β_i^{WRR-1}, β_i^{WRR-2} and β_i^{WRR-3}, Similar to β_i^{WFQ-1}, allocate bandwidth directly according to weights and packet sizes.

▼Table 1. Latency bounds for Strict Priority (SP) scheduling

Flow ID	Priority	τ_i/ms	σ_i/B	d_i/ms	Simulation Results/ms
1	High	1	256	0.409 6	0.256 0
2	Middle	1	256	0.579 5	0.406 8
3	Low	1	256	1.040 1	0.614 4

▼ Table 2. Flow configurations for Weighted Fair Queuing (WFQ) scheduling

Flow ID	Weight	τ_i/ms	σ_i/B
1	4	1	256
2	3	1	256
3	2	1	256

▼Table 3. Delay bounds for Weighted Fair Queuing (WFQ) scheduling

Method	d_1/ms	d_2/ms	d_3/ms
β_i^{WFQ-1}	0.576 0	0.768 0	1.152 0
β_i^{WFQ-2}	0.576 0	0.631 5	0.665 6
β_i^{WFQ-3}	0.576 0	0.631 5	0.665 6
Simulation results	0.406 8	0.512 0	0.614 4

▼Table 4. Flow configurations for RR scheduling

Scheduling Algorithm	Flow ID	Weight	τ_i/ms	σ_i/B
WRR	1	4	1	256
	2	3	1	256
	3	2	1	256
DRR	1	256	1	256
	2	192	1	256
	3	128	1	256

DRR: Deficit Round Robin RR: Round Robin WRR: Weighted Round Robin

▼Table 5. Delay bounds for RR scheduling

Scheduling Algorithm	Method	d_1/ms	d_2/ms	d_3/ms
WRR	β_i^{WRR-1}	0.716 8	0.921 6	1.280 0
	β_i^{WRR-2}	0.460 8	0.819 2	0.921 6
	β_i^{WRR-3}	0.460 8	0.819 2	0.921 6
	β_i^{BS-WRR}	0.716 8	0.815 5	0.870 6
	Simulation	0.406 8	0.614 4	0.614 4
IWRR	β_i^{IWRR}	0.460 8	0.665 6	0.921 6
	simulation	0.460 8	0.563 2	0.614 4
DRR	$\beta_i^{DRR-\varepsilon-1}$	0.716 8	0.921 6	1.280 0
	β_i^{BS-DRR}	0.460 8	0.614 4	0.921 6
	$\beta_i^{DRR-\varepsilon-2}$	0.460 8	0.819 2	0.921 6
	Simulation	0.406 8	0.614 4	0.614 4

DRR: Deficit Round Robin RR: Round Robin
IWRR: Interleaved Weighted Round Robin WRR: Weighted Round Robin

However, β_i^{WRR-2} and β_i^{WRR-3} utilize pseudo-inverse to leverage the details of RR (i.e. details of resource allocation), with β_i^{WRR-3} additionally considering packet-level details to implement tighter bounds. As the flows in this scenario are similar to periodic flows, β_i^{WRR-2} and β_i^{WRR-3} yield identical results. Especially for Flow 1, its burst packets are sent within a complete RR cycle in this scenario. During this cycle, other flows fully utilize their own transmission resources, resulting in an accurate boundary. β_i^{IWRR} employs the same derivation method as β_i^{WRR-2}, hence yielding similar results. However, IWRR alleviates bursts caused by continuous transmission of packets from the same flow, leading to a reduction in both the theoretical latency bounds and the maximum simulated latency for flow 2. On the other hand, β_i^{BS-WRR} uses resource utilization situations of other flows to improve the latency bounds compared to β_i^{WRR-1}. In certain scenarios, the results obtained are better than those of β_i^{WRR-2} and β_i^{WRR-3}.

The case is similar for DRR. $\beta_i^{DRR-\varepsilon-1}$ allocates resources directly, $\beta_i^{DRR-\varepsilon-2}$ utilizes pseudo-inverse to improve the bounds, and $\beta_i^{BS-DRR-1}$ considers the impact of cross-traffic to improve the bounds.

For CQF scheduling, the actual scheduling process involves the equivalence of two CQF queues. The odd or even queue depends on whether the corresponding queue's gate is open when data arrive. For an individual node, the upper bound on traffic flow delay is jointly determined by the delay bounds of data flows in both queues. However, for this case, the delay bounds are calculated separately for the two queues based on Eq. (49). This paper configures two flows with an alternating period of 4 ms. The different configurations and delay bounds for these business flows are presented in Table 6.

For TAS, this paper considers a simple scenario with three different flows. GLSs are configured based on the characteristics of the traffic flows. The queue gate opening period is the same as the flow period, and the GLS period is the least common multiple of all traffic flow periods. The gate control period is set to 1 ms. Based on the GCLs, the number of transmission windows for the three queues within one GCL period is 1, 2 and 2, respectively. Taking the start time of the GCL period as a reference, the waiting time S_i for the three queues is 0 ms, 2 ms and 2 ms, respectively. Based on Eq. (50), we can calculate the service provided to queue i within each transmission window and then derive the overall service curve. The flow configurations and their corresponding latency upper bounds are shown in Table 7.

In fact, the results in Table 7 do not represent the maximum latency of packets for each flow. This is because their service curves are referenced to a specific time point. In this paper, the starting time of the GCL period is taken as the reference point. The results reflect the maximum latency of all packets in the scenario where they arrive at the reference point. The differences in the results for different flows mainly arise from the time gap between the packet arrival time and the gate opening time.

On the other hand, ATS and CQF utilize gate structures to control transmission, ensuring precise allocation of transmission resources. With a comprehensive understanding of the flow characteristics, they can accurately calculate the latency bounds of packets, achieving deterministic transmission. Therefore, the configuration of the gate significantly influences the latency bounds of scheduling algorithms like CQF and ATS. In practical network scenarios, it is essential to set the configuration based on the characteristics of flows.

For the CBS scheduling, two SR flows are considered in this scenario: SR-A and SR-B. SR-A has a higher priority. Additionally, based on the settings in Refs. [10], [49] and [50], a CDT flow and a ST flow are introduced. In Refs. [49] and [50], the ST traffic arrival curve is determined by the GCLs without regard to the actual characteristics of the ST flows. In this paper, the ST queue's GCL period is set to 6 ms, with the gate opening for the first 1 ms of each period. The configuration and the corresponding delay bounds for each of these service flows are presented in Tables 8 and 9, respectively.

In the context of ATS, as discussed in Section 3.7, when multiple ATS nodes are connected in series, the shaper does not increase the overall system's delay bound. Therefore, this

▼ Table 6. Configurations and delay bounds for Round Robin (RR) scheduling

	τ_i/ms	σ_i/B	d_i/ms	Simulation Results/ms
β^{CQF-1}	1	256	0.204 8	0.204 8
β^{CQF-0}	1	256	4.204 8	4.204 8

▼Table 7. Configurations and delay bounds for TAS

Flow ID	GCL	τ_i/ms	σ_i/B	d_i/ms	Simulation Results/ms
1	100000	6	512	0.409 6	0.409 6
2	010010	3	521	1.409 6	1.409 6
3	001001	3	521	2.409 6	2.409 6

GCL: Gate Control List TAS: Time Aware Shaper

▼Table 8. Configurations for CBS

Flow ID	*idleSlope*/Mbit·s^{-1}	*sendSlope*/(Mbit·s^{-1})	τ_i/ms	σ_i/B
SR-A	4	−6	1	256
SR-B	3	−7	1	256
BE	-	-	1	256
CDT	-	-	1	64
ST	-	-	10	64

BE: best effort CDT: control data traffic ST: scheduled traffic
CBS: Credit Based Shaper SR: stream reservation

▼Table 9. Delay bound for CBS

		d_{SR-A}/ms	d_{SR-B}/ms	Simulation Results/ms	
				SR A	SR B
β_X^{CBS}		0.640 0	0.938 7	0.460 8	0.563 2
$\beta_X^{CBS-CDT}$		0.650 3	0.920 1	0.512 0	0.614 4
$\beta_i^{CBS-ST-1}$	P	1.588 2	1.852 5	1.470 8	1.573 2
	NP	1.614 4	1.870 4	1.512 0	1.614 4
$\beta_i^{BS-ST-2}$	F	1.614 4	1.870 4	1.512 0	1.614 4
	NF	1.615 3	1.907 7	1.460 8	1.563 2

CBS: Credit Based Shaper NF: non-frozen P: preemptive
F: frozen NP: non-preemptive SR: stream reservation

paper only considers delay $D_{Q_i}^{ATS}$ for the shared queue. The upper bound of the delay is determined by the output arrival curve of shaper $\alpha_q^*(t)$ and the service curve of shared queue $\beta_{Q_i}^{ATS}(t)$. This paper examines three different priority data flows, each entering a different shaper. The relevant parameters and the upper bounds of delay for these flows are presented in Table 10.

5 Conclusions

In this paper, an overview of seven common scheduling algorithms is provided. We summarize the service curves of these scheduling algorithms in different implementations (preemptive and non-preemptive) and under various traffic categories based on existing literature related to network calculus and QoS analysis. Finally, this paper applies different service curves to calculate their corresponding delay bounds in burst flow situations and conducts simulations in corresponding situations. Additionally, this paper conducts a comparative analysis between the simulation results and the theoretical outcomes. The study of this paper can serve as a reference for further research in the field of network modeling and QoS analysis using network calculus.

▼Table 10. Configurations and delay bounds for ATS

Flow ID	Committed Transmission Rate/(Mbit·s^{-1})	Burst Size/B	τ_i/ms	σ_i/B	d_i/ms
1	4	128	1	256	0.256 0
2	3	128	1	256	0.597 3
3	2	128	1	256	1.563 0

References

[1] IETF. Integrated services in the internet architecture: an overview: RFC 1633 [S]. 1994
[2] IETF. An architecture for differentiated services: RFC2475 [S]. 1998
[3] DON W. The history of the IEEE 802 standard [J]. IEEE communications standards magazine, 2018, 2(2): 4. DOI: 10.1109/MCOMSTD.2018.8412452
[4] IETF. Deterministic networking problem statement: RFC8557 [S]. 2019
[5] HE F, ZHAO L, LI E S. Impact analysis of flow shaping in ethernet-AVB/TSN and AFDX from network calculus and simulation perspective [J]. Sensors, 2017, 17(5): 1181. DOI: 10.3390/s17051181
[6] FINZI A, MIFDAOUI A, FRANCES F, et al. Incorporating TSN/BLS in AFDX for mixed-criticality applications: model and timing analysis [C]//Proc. 14th IEEE International Workshop on Factory Communication Systems (WFCS). IEEE, 2018: 1 – 10. DOI: 10.1109/WFCS.2018.8402346
[7] ZHAO L X, POP P, STEINHORST S. Quantitative performance comparison of various traffic shapers in time-sensitive networking [J]. IEEE transactions on network and service management, 2022, 19(3): 2899 – 2928. DOI: 10.1109/TNSM.2022.3180160
[8] SPECHT J, SAMII S. Urgency-based scheduler for time-sensitive switched Ethernet networks [C]//Proc. 28th Euromicro Conference on Real-Time Systems (ECRTS). IEEE, 2016: 75 – 85. DOI: 10.1109/ECRTS.2016.27
[9] LE BOUDEC J Y. A theory of traffic regulators for deterministic networks with application to interleaved regulators [J]. IEEE/ACM transactions on networking, 2018, 26(6): 2721 – 2733. DOI: 10.1109/TNET.2018.2875191
[10] MOHAMMADPOUR E, STAI E, MOHIUDDIN M, et al. Latency and backlog bounds in time-sensitive networking with credit based shapers and asynchronous traffic shaping [C]//Proc. 30th International Teletraffic Congress (ITC 30). IEEE, 2018: 1 – 6
[11] JIANG Y M. Some properties of length rate quotient shapers [EB/OL]. (2021-07-11)[2023-10-13]. http://arxiv.org/abs/2107.05021
[12] JIANG Y M. A basic result on the superposition of arrival processes in deterministic networks [C]//Proc. IEEE Global Communications Conference (GLOBECOM). IEEE, 2018: 1 – 6. DOI: 10.1109/GLOCOM.2018.8647202
[13] CRUZ R L. A calculus for network delay. part I: network elements in isolation [J]. IEEE transactions on information theory, 1991, 37(1): 114 – 131. DOI: 10.1109/18.61109
[14] CRUZ R L. A calculus for network delay, part II: network analysis [J]. IEEE transactions on information theory, 1991, 37(1): 132 – 141. DOI: 10.1109/18.61110
[15] PAREKH A K, GALLAGER R G. A generalized processor sharing approach to flow control in integrated services networks: the single-node case [J]. IEEE/ACM transactions on networking, 1993, 1(3): 344 – 357. DOI: 10.1109/90.234856
[16] PAREKH A K, GALLAGER R G. A generalized processor sharing approach to flow control in integrated services networks: the multiple node case [J]. IEEE/ACM transactions on networking, 1994, 2(2):137 – 150. DOI: 10.1109/90.234856
[17] CRUZ R L. Quality of service guarantees in virtual circuit switched networks [J]. IEEE journal on selected areas in communications, 1995, 13(6): 1048 – 1056. DOI: 10.1109/49.400660
[18] SARIOWAN H, CRUZ R L, POLYZOS G C. Scheduling for quality of service guarantees via service curves [C]//Proc. Fourth International Conference on Computer Communications and Networks. IEEE, 1995: 512 – 520. DOI: 10.1109/ICCCN.1995.540168
[19] AGRAWAL R, RAJAN R. Performance bounds for guaranteed and adaptive services: IBM Technical Report RC 20649 [R]. 1996
[20] CRUZ R L. SCED: efficient management of quality of service guarantees [C]// Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE, 1998: 625 – 634. DOI: 10.1109/INFCOM.1998.665083
[21] LE BOUDEC J Y. Application of network calculus to guaranteed service networks [J]. IEEE transactions on information theory, 1998, 44(3): 1087 – 1096. DOI: 10.1109/18.669170
[22] CHANG C S. On deterministic traffic regulation and service guarantees: a systematic approach by filtering [J]. IEEE transactions on information theory, 1998, 44(3): 1097 – 1110. DOI: 10.1109/18.669173
[23] AGRAWAL R, CRUZ R L, OKINO C, et al. Performance bounds for flow

control protocols [J]. IEEE/ACM transactions on networking, 1999, 7(3): 310 – 323. DOI: 10.1109/90.779197
[24] FIDLER M. Survey of deterministic and stochastic service curve models in the network calculus [J]. IEEE communications surveys & tutorials, 2010, 12 (1): 59 – 86. DOI: 10.1109/SURV.2010.020110.00019
[25] LE BOUDEC J Y, THIRAN P. Network calculus: a theory of deterministic queuing systems for the Internet [M]. Berlin: Springer, 2001
[26] STILIADIS D, VARMA A. Latency-rate servers: a general model for analysis of traffic scheduling algorithms [J]. IEEE/ACM transactions on networking, 1998, 6(5): 611 – 624. DOI: 10.1109/90.731196
[27] JIANG Y M, LIU Y. Stochastic network calculus [M]. London: Springer, 2008
[28] BURCHARD A, LIEBEHERR J. A general per-flow service curve for GPS [C]//Proc. 30th International Teletraffic Congress (ITC 30. IEEE, 2018: 31 – 36
[29] NAGLE J. On packet switches with infinite storage [J]. IEEE transactions on communications, 1987, 35(4): 435 – 438. DOI: 10.1109/TCOM.1987.1096782
[30] KATEVENIS M, SIDIROPOULOS S, COURCOUBETIS C. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip [J]. IEEE journal on selected areas in communications, 1991, 9(8): 1265 – 1279. DOI: 10.1109/49.105173
[31] SHREEDHAR M, VARGHESE G. Efficient fair queuing using deficit round-robin [J]. IEEE/ACM transactions on networking, 1996, 4(3): 375 – 385. DOI: 10.1109/90.502236
[32] BOUILLARD A, BOYER M, LE CORRONC E. Deterministic network calculus: from theory to practical implementation [M]. Hoboken: Wiley, 2018. DOI: 10.1002/9781119440284
[33] TABATABAEE S M, LE BOUDEC J Y, BOYER M. Interleaved weighted round-robin: a network calculus analysis [J]. IEICE Transactions on Communications, 2021, 104(12): 1479 – 1493. 10.1587/TRANSCOM.2021ITI0001
[34] KANHERE S S, SETHU H. On the latency bound of deficit round robin [C]// Proc. Eleventh International Conference on Computer Communications and Networks. IEEE, 2002: 548 – 553. DOI: 10.1109/ICCCN.2002.1043123.
[35] LENZINI L, MINGOZZI E, STEA G. Full exploitation of the deficit round Robin capabilities by efficient implementation and parameter tuning [R]. 2003
[36] BOYER M, STEA G, MANGOUA SOFACK W. Deficit round robin with network calculus [C]//Proc. 6th International Conference on Performance Evaluation Methodologies and Tools. IEEE, 2012: 138 – 147. DOI: 10.4108/valuetools.2012.250202
[37] BOUILLARD A. Individual service curves for bandwidth-sharing policies using network calculus [J]. IEEE networking letters, 2021, 3(2): 80 – 83. DOI: 10.1109/LNET.2021.3067766
[38] TABATABAEE S M, LE BOUDEC J Y. Deficit round-robin: a second network calculus analysis [J]. IEEE/ACM transactions on networking, 2022, 30 (5): 2216 – 2230. DOI: 10.1109/TNET.2022.3164772
[39] CONSTANTIN V C, NIKOLAUS P, SCHMITT J. Improving performance bounds for weighted round-robin schedulers under constrained cross-traffic [C]//Proc. IFIP Networking Conference (IFIP Networking). IEEE, 2022: 1 – 9
[40] IEEE. IEEE standard for local and metropolitan area networks: bridges and bridged networks: amendment 29: cyclic queuing and forwarding: 802.1Qch-2017 [S]. 2017
[41] YIN S W , WANG S , HUANG T . Analysis and optimization of queues based on network calculus in time-sensitive networking [J]. ZTE technology journal, 2022, 28(1): 21 – 28. DOI:10.12142/ZTETJ.202201007
[42] WANDELER E, THIELE L. Optimal TDMA time slot and cycle length allocation for hard real-time systems [C]//Proc. Asia and South Pacific Conference on Design Automation. IEEE, 2006: 479 – 484. DOI: 10.1109/ASPDAC.2006.1594731
[43] IEEE. IEEE standard for local and metropolitan area networks: bridges and bridged networks: amendment 25: enhancements for scheduled traffic: 802.1Qbv-2015 [S]. 2015
[44] IEEE. IEEE standard for local and metropolitan area networks: bridges and bridged networks: amendment 26: frame preemption: 802.1Qbu-2016 [S]. 2016
[45] ZHAO L X, POP P, CRACIUNAS S S. Worst-case latency analysis for IEEE 802.1Qbv time sensitive networks using network calculus [J]. IEEE access, 2018, 6: 41803 – 41815. DOI: 10.1109/ACCESS.2018.2858767
[46] IEEE. IEEE standard for local and metropolitan area networks: virtual bridged local area networks amendment 12: forwarding and queuing enhancements for time-sensitive streams: 802.1Qav [S]. 2009
[47] DE AZUA J A R, BOYER M. Complete modelling of AVB in Network Calculus Framework [C]//Proc. 22nd International Conference on Real-Time Networks and Systems. ACM, 2014: 55 – 64. DOI: 10.1145/2659787.2659810
[48] ZHAO L, HE F, LI E S, et al. Improving worst-case delay analysis for traffic of additional stream reservation class in ethernet-AVB network [J]. Sensors, 2018, 18(11): 3849. DOI: 10.3390/s18113849
[49] ZHAO L X, POP P, ZHENG Z, et al. Timing analysis of AVB traffic in TSN networks using network calculus [C]//Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE, 2018: 25 – 36. DOI: 10.1109/RTAS.2018.00009
[50] ZHAO L X, POP P, ZHENG Z, et al. Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus [J]. IEEE transactions on industrial electronics, 2020, 68(10): 10291 – 10302. DOI: 10.1109/TIE.2020.3021638
[51] IEEE. IEEE standard for local and metropolitan area networks-bridges and bridged networks amendment 34: asynchronous traffic shaping: 802.1 Qcr-2020 [S]. 2020

Biographies

GAO Yuehong (yhgao@bupt.edu.cn) received her PhD degree from Beijing University of Posts and Telecommunications (BUPT), China in 2010. She is an associate professor with the School of Information and Communication Engineering, BUPT. Her research interests include network calculus theory and application, quality of service guarantees in communication networks, simulation methodology, and digital twin networks.

NING Zhi received his BE degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), China in 2021. He is working towards his MS degree at BUPT. His research interests include wireless networks and deterministic networking.

HE Jia received his BE degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), China in 2022. He is working towards his MS degree in communication engineering at BUPT. His research interests include network calculus, 5G network architecture, and deterministic networking.

ZHOU Jinfei received his BE degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), China in 2023, where he is currently pursuing an MS degree. His research interests include wireless networks and deterministic networking.

GAO Chenqiang works at in the Data System Department of ZTE Corporation. His research interests include network calculus theory and application, DetNet, TSN, and SDN.

TANG Qingkun works at the Cable Software Platform Development Department of ZTE Corporation. His research interests include DetNet, TSN, and autonomous networks.

YU Jinghai works at the Data System Department of ZTE Corporation. He has more than 20 years of experience in the research and design of data network products including BIER, DetNet, TSN, Switch and Router, Data Center and SDN. He has won the 21st China Patent Silver Award and the first prize of the Science and Technology Award of the China Communications Society.