Postprint of Applied Research on Improved WFQ Algorithm in Train Network Scheduling

Authors: Qin Jianxiu, Song Jianlin, Dai Xiaowen

Date: 2018-08-13T00:00:00+00:00

Abstract

With the rapid development of rail transit in China, TCN-based train networks can no longer satisfy the demands of train data transmission services, giving rise to the Train Control and Service Network (TCSN). This paper investigates train communication networks based on the Integrated Services model for train data transmission services, establishes four categories of train service traffic models, analyzes the Weighted Fair Queuing (WFQ) scheduling algorithm for network packets, and designs a Service Classification and Bandwidth Allocation-based Weighted Fair Queuing scheduling algorithm (SCBAWFQ). Furthermore, simulation software is employed to simulate and analyze the delay characteristics of both constant and bursty data streams under WFQ and SCBAWFQ algorithms. The simulation results demonstrate that the improved algorithm effectively reduces the delay of real-time service flows, thereby enabling the SCBAWFQ algorithm to maintain the fairness of WFQ while satisfying the Quality of Service (QoS) requirements of real-time services.

Full Text

Preamble

Title: Research on Application of Improved WFQ Algorithm in Train Network Scheduling

Authors: Qin Jianxiu, Song Jianlin, Dai Xiaowen
Affiliation: School of Electrical Engineering, Southwest Jiaotong University, Chengdu 611756, China

Abstract: With the rapid development of rail transit in China, TCN-based train networks can no longer meet the demands of train data transmission services, giving rise to the Train Control and Service Network (TCSN). This paper investigates train communication networks based on the integrated service model for train data transmission services, establishes four types of train traffic flow models, analyzes the Weighted Fair Queueing (WFQ) scheduling algorithm for network packets, and designs a Service Classification and Bandwidth Allocation-based Weighted Fair Queueing (SCBAWFQ) algorithm. Using simulation software, we simulate and analyze the delay of constant data streams and bursty data streams under both WFQ and SCBAWFQ algorithms. Simulation results demonstrate that the improved algorithm effectively reduces delay for real-time service flows, thereby enabling SCBAWFQ to maintain the fairness of WFQ while satisfying the quality-of-service requirements of real-time services.

Keywords: train control and service network; scheduling algorithm; SCBAWFQ algorithm; network simulation

0 Introduction

With the rapid development of train communication technology in China and the improvement of people' s living standards, train networks can no longer merely guarantee the transmission of train control service data; they must also provide passenger service information, such as Internet access for entertainment. Industrial Ethernet has demonstrated significant advantages in various applications, leading many researchers to propose its introduction into train networks to solve the bandwidth limitations of traditional train networks. However, train network performance depends not only on network topology and communication protocols but also on efficient scheduling algorithms that maximize network resource utilization and meet the real-time and reliability requirements of train network systems.

1 Current Status of Train Networks and Scheduling Algorithms

In recent years, China' s rapid development in high-speed railways and high-speed trains has imposed higher demands on safety, reliability, and efficiency, while also requiring continuous improvement in high-speed train performance, comfort, and intelligence. Traditional train networks mainly include WorldFIP, LonWorks, CAN, ARCNET, and TCN networks, each with certain deficiencies such as poor real-time performance, low reliability, and insufficient communication bandwidth. With the integration of industrial Ethernet and train networks, the International Electrotechnical Commission (IEC) has introduced relevant standards for Ethernet-based train communication networks, significantly advancing train network development and leading to many improved train network architectures. The Train Control and Service Network (TCSN) has emerged from the early Train Communication Network (TCN) with the goal of establishing a train network featuring large bandwidth, high reliability, high trustworthiness, reconfigurability, and intelligence.

Scheduling algorithms are resource allocation algorithms defined according to system resource allocation policies and represent a core technology for implementing network Quality of Service (QoS). Through different network scheduling strategies, network resource allocation can be optimized to improve resource utilization and avoid network congestion. Key metrics for evaluating packet scheduling algorithms include delay performance, fairness, simplicity, scalability, and complexity. Common scheduling algorithms include Priority Queue (PQ), Queue Length Threshold (QLT), Earliest Deadline First (EDF), and Weighted Fair Queueing (WFQ).

Domestic and international scholars have conducted extensive research in this area. Reference [10] proposes a Grouped Tasks Scheduling (GTS) algorithm that classifies all tasks into five categories based on four characteristics: user type, task type, task size, and task delay. The five categories are scheduled according to priority, and within each category, tasks are scheduled using the Min-Min algorithm. The GTS algorithm prioritizes category scheduling before task scheduling, which largely guarantees the priority of urgent tasks and tasks with minimum execution time, effectively reducing their delay but potentially increasing delay for long-duration tasks.

Reference [11] proposes a fuzzy logic-based scheduling method that develops a fuzzy expert system. This system can recalculate task weight coefficients based on changes in task queue length, throughput, and delay characteristics, achieving optimized resource allocation and reasonable scheduling by autonomously adjusting the weight coefficients of real-time and non-real-time business queues. This algorithm effectively guarantees the quality of service for both real-time and non-real-time services under heavy load conditions, but the system is relatively complex, and fuzzy logic design and optimization require considerable time. Alternative optimization methods could be explored, such as neural networks.

Reference [12] proposes the FC-WFQ algorithm, which dynamically adjusts the forwarding bandwidth weights of various data flows in real-time according to command arrival rates and read/write ratios within commands. Experimental results show that this scheduling algorithm can significantly reduce the average response time of read/write tasks and substantially improve throughput. This algorithm no longer marks the physical forwarding length of data packets but instead marks their logical forwarding length based on the processor resources required to forward each packet. As load conditions intensify, the FC-WFQ algorithm' s advantage in rational allocation of forwarding bandwidth becomes more pronounced.

Reference [13] proposes a Global Preemptive Threshold Scheduling (GPTS) algorithm for real-time systems, a hybrid scheduling approach between preemptive and non-preemptive scheduling. This algorithm sets a preemptive threshold for each task, which increases the task' s priority after it begins execution, reducing the possibility of preemption by higher-priority tasks. While maintaining system schedulability, this significantly reduces the number of preemptions during system operation, decreases resource consumption, and improves scheduling capability.

Reference [14] addresses periodic message delays in CAN networks by proposing a static scheduling algorithm based on network node priority. Building upon the monotonic rate scheduling algorithm, this approach analyzes the relationship between network node priority and message periods to identify the time point with maximum communication delay, which is then introduced as a delay variable into the mathematical model. This enables accurate estimation of network node delay. Simulation verification demonstrates that the algorithm can effectively reduce network delay.

2 WFQ Algorithm

The WFQ scheduling algorithm [15] classifies packets based on flows, assigning flows with the same priority to the same queue to ensure fairness for all data packets within that queue. During packet dequeuing, WFQ allocates bandwidth to each queue according to its weight value. Providing as many queues as possible enables more uniform distribution of each flow into different queues to satisfy the delay characteristics of various flow services. The weight is calculated from the priority of flows in the corresponding queue.

WFQ introduces two new concepts: virtual clock and virtual time function. The virtual time function is a function that changes over time. Let t_j denote the time when the j-th event occurs, $B(\tau)$ denote the set of all backlogged sessions during the interval (t_{j-1}, t_j), and ϕ_i denote a positive real number representing session i, also known as the weight or service share parameter (reserved bandwidth size) of each session. The definition of virtual time $V(t)$ is given by Equation (1):

$$V(t) = V(t_{j-1}) + \frac{t - t_{j-1}}{\sum_{i \in B_j} \phi_i}, \quad t_{j-1} < t \leq t_j, \quad j = 1, 2, 3, ...$$

Building upon this definition, the WFQ algorithm also defines two variables—virtual start time and virtual finish time—which are assigned to packets arriving at the queue. Let P_i^k denote the k-th packet of connection i, a_i^k denote its arrival time, S_i^k denote its start service time tag, F_i^k denote its finish service time tag, and L_i^k denote its packet length. Let r_i denote the service rate reserved by connection i, satisfying $r_i \geq 0$ for all sessions. Therefore, the virtual start time S_i^k and virtual finish time F_i^k of packet P_i^k are calculated using Equations (2) and (3):

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

$$F_i^k = S_i^k + \frac{L_i^k}{r_i}$$

The scheduling rule of the WFQ algorithm is Smallest Finish-time First (SFF), meaning the packet with the smallest virtual finish time is transmitted first. The WFQ algorithm in this paper is constrained by a leaky bucket, and its end-to-end delay bound is given by Equation (4):

$$\text{Delay} \leq \frac{\sigma_i}{r_i} + \frac{L_{\max}}{R}$$

where σ_i represents the maximum end-to-end delay jitter for data transmission, $L_{\max}$ represents the maximum packet length, and R represents the server service rate.

WFQ can precisely control individual service flows and guarantee the end-to-end delay for each flow. However, WFQ has excessively fine computational granularity, resulting in high computational complexity, especially as the number of packet queues increases. Additionally, it does not handle well traffic with variable lengths and burstiness. Specifically, WFQ has the following shortcomings in practical applications:

a) WFQ cannot distinguish between service types. This occurs because WFQ is fundamentally a fair scheduling algorithm that only calculates the minimum virtual time for packets in each flow during scheduling without considering priority differences between packets, causing all sessions to share bandwidth resources proportionally according to their weights.

b) WFQ has bandwidth allocation deficiencies. Since WFQ allocates bandwidth resources proportionally according to weights, it is often difficult to guarantee quality of service for bursty traffic flows.

c) WFQ exhibits coupling between bandwidth guarantee and delay guarantee, meaning low bandwidth guarantee implies weak or reduced delay guarantee. This deficiency makes WFQ unsuitable for scheduling certain types of traffic, such as voice and video services, which require small bandwidth but strict delay guarantees.

3 Improved WFQ Algorithm

WFQ does not classify sessions in detail, meaning all sessions have the same service priority. Therefore, this paper designs and implements an improved weighted fair queueing scheduling algorithm based on WFQ called Service Classification and Bandwidth Allocation-based WFQ (SCBAWFQ).

3.1 Iterative Bandwidth Allocation Method

Assume there are P sessions in a network node, each session i has a corresponding priority ρ_i, and each session has a weight value w_i. The priority is assigned based on the transmission volume and importance of each service type. The weight w_i is calculated using Equation (5):

$$w_i = \frac{1}{1+\rho_i}$$

Let L_i denote the size of each packet and λ_i denote the packet arrival rate. The product of these two variables represents the reserved bandwidth size for the current session, as shown in Equation (6):

$$I_i = L_i \times \lambda_i$$

SCBAWFQ first determines the service type, distinguishing between real-time and non-real-time services. Real-time services are allocated bandwidth equal to their reserved bandwidth to ensure their quality of service. The remaining bandwidth is then calculated and allocated to all non-real-time services according to the iterative bandwidth allocation principle. The bandwidth allocation is given by Equation (11), where S represents the set of real-time service sessions and $\overline{S}$ represents the set of non-real-time service sessions.

The iterative bandwidth allocation method proceeds as follows. Let B represent the total link bandwidth. In the first iteration, the bandwidth allocated to each session may result in two scenarios: first, each session receives just enough bandwidth for transmission or less than required; second, each session receives more bandwidth than required. The first iteration can be expressed using Equation (7):

$$B_i^1 = \min\left(w_i \times B, I_i\right), \quad i = 1, 2, \ldots, P$$

If, after the first iteration, some sessions receive more than their reserved bandwidth while others receive less, the excess bandwidth from some sessions will be redistributed to others through subsequent iterations. The excess bandwidth is allocated to other sessions proportionally according to their weights, using Equation (8) to determine whether each session needs additional bandwidth:

$$\Delta_i = I_i - B_i^k$$

If Equation (8) yields 0, session i has received its required bandwidth. If the result is a positive real number, session i received less than its reserved bandwidth in that iteration. In the worst case, bandwidth allocation for sessions requires $P-1$ iterations to redistribute excess bandwidth from each iteration to other sessions until no excess bandwidth remains. Each iteration process is given by Equation (9):

$$B_i^{k+1} = B_i^k + \frac{w_i}{\sum_{j\in Q_k} w_j} \times \sum_{j\in Q_k}\left(I_j - B_j^k\right)$$

where Q_k represents the set of sessions that have not yet received sufficient bandwidth in the k-th iteration.

3.2 SCBAWFQ Algorithm Principle

The SCBAWFQ algorithm marks priority for each session' s service. When total network bandwidth is insufficient, the algorithm can guarantee the quality of service for high-priority real-time services while maintaining fairness comparable to WFQ. The bandwidth allocation process uses the iterative bandwidth allocation method as follows:

1) **Sufficient total network bandwidth** ($\sum_{i=1}^{P} I_i \leq R$):
Let I_i represent the reserved bandwidth size for session i. The SCBAWFQ algorithm can satisfy the quality of service for all sessions, so it directly adopts the iterative allocation model, with bandwidth allocation given by Equation (10):

$$B_i = \min\left(w_i \times R, I_i\right)$$

2) **Insufficient total network bandwidth** ($\sum_{i=1}^{P} I_i > R$):
The first iteration may produce two scenarios. In the first scenario, each session receives just enough bandwidth for transmission or less than required. In the second scenario, each session receives more bandwidth than required. The first iteration can be expressed using Equation (7). The SCBAWFQ algorithm first identifies the service type, distinguishing between real-time and non-real-time services. Real-time services are allocated bandwidth equal to their reserved bandwidth to ensure their quality of service. The remaining bandwidth is then allocated to all non-real-time services according to the iterative bandwidth allocation principle, as shown in Equation (11):

$$B_i = \begin{cases} I_i, & i \in S \\ \min\left(w_i \times \left(R - \sum_{j \in S} I_j\right), I_i\right), & i \in \overline{S} \end{cases}$$

The flowchart of the SCBAWFQ scheduling algorithm is shown in Figure 1 [Figure 1: see original paper].

The main differences between SCBAWFQ and WFQ are: (1) SCBAWFQ classifies sessions by service type, and (2) SCBAWFQ uses an iterative bandwidth allocation scheme to enable more reasonable bandwidth distribution for each session. Through service classification, SCBAWFQ overcomes WFQ' s deficiency of proportional bandwidth allocation. The iterative bandwidth allocation method enables more complete and rational bandwidth utilization. When traffic volume exceeds total bandwidth resources, SCBAWFQ prioritizes real-time services to ensure their quality of service requirements, while still satisfying fairness requirements for non-real-time services to a certain extent.

4 Simulation Results Analysis

We establish four types of traffic flows conforming to the leaky bucket model to analyze and verify the delay performance of these four services under both WFQ and SCBAWFQ algorithms. The network topology is shown in Figure 2 [Figure 2: see original paper]. The source node side of the network topology consists of four source nodes, each capable of generating corresponding data flows representing traction and braking control services, auxiliary power and air conditioning control services, video passenger service information, and ordinary data passenger services. All four types of traffic flows use the UDP protocol, flowing through the node and eventually reaching the terminal.

4.1 Constant Traffic Flow Simulation

The simulation bandwidth is set to 90 Mbps, with parameters shown in Table 1 . Here, D represents the maximum end-to-end delay, σ represents the maximum burst size of the traffic flow, r represents the maximum rate of traffic generation, and w represents the weight corresponding to the data flow. The first three traffic flows are configured as real-time services, and the fourth as non-real-time service.

Table 1: SCBAWFQ Algorithm Simulation Parameter Settings

Traffic Type	L (byte)	λ (Mbps)	I (Mbps)	ρ	w	D (ms)	σ (byte)	r (Mbps)
Traction/Braking	100	0.2	0.8	10	0.091	10	100	1
Power/Air Conditioning	100	0.8	3.2	8	0.111	50	200	5
Video	500	0.8	4.0	6	0.143	100	500	10
Data	1000	0.2	1.6	4	0.2	200	1000	20

Figures 3 [Figure 3: see original paper] and 4 [Figure 4: see original paper] show the end-to-end delay simulation results for the four services under the two scheduling algorithms. In the figures, curve represents traction and braking services, curve represents auxiliary power and air conditioning services, curve represents video services, and curve represents data services.

Figure 3: End-to-End Delay Simulation under WFQ Algorithm
Figure 4: End-to-End Delay Simulation under SCBAWFQ Algorithm

Data analysis from the figures shows that real-time data flows 1, 2, and 3 under SCBAWFQ exhibit significant delay reductions compared to WFQ, decreasing by 25 ms, 165 ms, and 107 ms respectively. This demonstrates that SCBAWFQ substantially improves delay performance for real-time traction and braking services, auxiliary power and air conditioning services, and video services. Non-real-time data flow 4 has a delay of 154 ms under WFQ and 158 ms under

SCBAWFQ. Both delays meet the 200 ms requirement, but SCBAWFQ' s delay is slightly higher because SCBAWFQ sacrifices a small portion of non-real-time service end-to-end delay performance to guarantee real-time service delay performance.

4.2 Bursty Traffic Flow Simulation

The simulation bandwidth is set to 90 Mbps, with parameters as shown in Table 1. Figures 5 [Figure 5: see original paper] and 6 [Figure 6: see original paper] present the end-to-end delay simulation results for the four services under WFQ and SCBAWFQ algorithms respectively. The curve labels remain the same as in the constant traffic case.

Figure 5: End-to-End Delay Simulation under WFQ Algorithm
Figure 6: End-to-End Delay Simulation under SCBAWFQ Algorithm

Analysis of the two algorithms' simulation results reveals:

Curve shows that under WFQ, delay fluctuates between 31-33 ms, far exceeding the service requirement. Under SCBAWFQ, delay fluctuates between 6-7.5 ms during simulation seconds 2.5-4 due to bursts in all three services causing a significant increase in total traffic volume, but remains stable at 6 ms for the remaining time, overall meeting the delay requirement.

Curve shows that under WFQ, normal transmission delay is approximately 67 ms, but increases significantly during bursts, reaching a maximum of about 200 ms, potentially exceeding service requirements. Under SCBAWFQ, delay is approximately 35 ms, with fluctuations between 35-50 ms during seconds 2.5-4 due to the substantial increase in total traffic. Overall, the improved algorithm significantly reduces delay for this real-time service and meets its end-to-end delay requirement.

Curve shows that under WFQ without bursts, delay is 93 ms, but increases to a maximum of 127 ms after bursts, exceeding the 100 ms requirement. Under SCBAWFQ, delay remains at 93 ms. The improved algorithm reduces video service delay and satisfies the real-time service delay requirement throughout the entire process.

Curve represents non-real-time service. Under WFQ, delay increases slowly in the first 2 seconds, then rises rapidly after second 2 due to data bursts, reaching a maximum of 75.4 ms. Under SCBAWFQ, delay increases at a certain rate to 76.5 ms. Both delays meet the 200 ms requirement, but SCBAWFQ' s delay is slightly higher, confirming that SCBAWFQ sacrifices non-real-time service end-to-end delay performance.

5 Conclusion

This paper proposes the Service Classification and Bandwidth Allocation-based Weighted Fair Queueing (SCBAWFQ) scheduling algorithm. The algorithm first classifies service types to distinguish between real-time and non-real-time services, then employs an iterative bandwidth allocation scheme to allocate bandwidth to each service, ensuring quality of service for real-time services with bursty traffic. Simulation results demonstrate that for both constant and bursty data streams, SCBAWFQ maintains the same fairness as WFQ while reducing packet delay for all real-time services and guaranteeing their quality of service, albeit at the cost of slightly sacrificing non-real-time service quality.

References

[1] Zhang Hua, Tan Xianhai, Zhao Jinnan, et al. Scheduling policy for traffic of TCSN [J]. Railway Computer Application, 2017, 26 (11): 54-58.

[2] Zhuge Xuanyu, Guo Qiyi, Wang Mengchao. Research on reliability evaluation method of train communication network [J]. Electric Drive for Locomotives, 2017 (4): 9-14.

[3] Guo Qiyi, Kang Jinsong, Feng Jianghua, et al. Current development and trend of rail transit vehicle traction control [J]. Journal of Power Supply, 2017, 15 (02): 40-45.

[4] Chen Jiakai. Research on optimization of communication mode on train communication network and reliability analysis [D]. Hangzhou: Zhejiang University, 2015.

[5] Zhang Hua, Tan Xianhai, Zhao Jinnan, et al. TCSN scheduling algorithm for dynamic adjustment scheduling quotas [J/OL]. Application Research of Computers, 2018, 35 (11). [2018-06-02]. http://kns.cnki.net/kcms/detail/51.1196.TP.20171115.1041.028.html.

[6] Bai Xue, Tan Xianhai, Liu Lihao. High-speed train control and service network (tcsn) business analysis and modeling [J]. Internet of Things Technologies, 2017, 7 (8): 84-86.

[7] Li Xiangning, Ma Jing. Research on modeling and simulation of data services of TCN [C]// Proc of the 2nd International Conference on Computer and Automation Engineering. 2010.

[8] Fan Zhaoyong, Gan Zhichun, Shen Jianjun, et al. Research on networking simulation in TCN simulation training [C]// Proc of International Asia Conference on Informatics in Control, Automation and Robotics. 2009.

[9] Luo Jie. Train network applied research based on high-speed ethernet train backbone network [D]. Chengdu: Southwest Jiaotong University, 2013.

[10] El Din Hassan Ali H G, Saroit I A, Kotb A M. Grouped tasks scheduling algorithm based on QoS in cloud computing network [J]. Egyptian Informatics Journal, 2017, 18 (1): 11-19.

[11] Akashdeep, Kahlon KS. An embedded fuzzy expert system for adaptive WFQ scheduling of IEEE 802.16 networks [J]. Expert Systems with Applications, 2014, 41 (16): 7621-7629.

[12] Han Yong, Yao Nianmin, Cai Shaobin, et al. Packet forwarding scheduling algorithm FC-WFQ for iSCSI virtualization switch [J]. Acta Electronica Sinica, 2013, 41 (03): 587-592.

[13] Peng Hao. Scheduling algorithms for multiprocessor hard real-time systems [D]. Hefei: Hefei University of Technology, 2015.

[14] Chen Anyu. The research and implementation of scheduling algorithm in passenger vehicle network control system [D]. Wuhan: Wuhan University, 2014.

[15] Li Bingquan, Zhang Song, Wang Zhaowei, et al. Performance analysis and improvement about WFQ and WRR schedule algorithm [J]. Transactions of Beijing Institute of Technology, 2015, 35 (3): 316-320.

[16] Li Changxian, Zhao Yan, Zhang Tong, et al. Research on Delay Characteristics of Train Network Control System [J]. China Railway, 2018 (1): 49-53, 59.

Note: Figure translations are in progress. See original paper for figures.