INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 19, Issue: 1, Month: Feburary, Year: 2024
Article Number: 5962, https://doi.org/10.15837/ijccc.2024.1.5962

CCC Publications

Optimized QoS Routing in Software-Defined In-Vehicle Networks

Y.H. Zhai, Y. Lv, L.Y. Xu*

Yahong Zhai
School of Electrical & Information Engineering
Hubei University of Automotive Technology
Shiyan 442002, China
20070026@huat.edu.cn

Yong Lv
School of Electrical & Information Engineering
Hubei University of Automotive Technology
Shiyan 442002, China
201911020@huat.edu.cn

Longyan Xu*
School of Electrical & Information Engineering
Hubei University of Automotive Technology
Shiyan 442002, China
*Corresponding author: 20060018@huat.edu.cn

Abstract

To address the problems of low network centralized management, weak interaction, limited hardware scalability, compatibility issues and challenges in expansion in the static deployment of traditional in-vehicle networks (IVNs), a new IVN architecture is designed. At the same time, to better meet the IVN Quality of Service (QoS) requirements and improve the real-time guarantee of data transmission, the QoS routing optimization mechanism under the new architecture is established. First, a new IVN architecture including a forwarding plane, control plane and application plane is designed by introducing software defined network (SDN) technology and combining it with the IVN itself. Second, the end-to-end delay optimization model of IVN is established by introducing network calculus theory, defining system parameters, creating a network model, calculating latency, considering queuing and congestion. the traditional routing algorithm is improved, and the DBROA algorithm has been proposed, which enhances the performance of QoS routing by introducing features such as distributed routing decisions, beacon mechanisms, optimization algorithms, and adaptability. This improvement allows it to better meet the QoS requirements of various applications and services, thereby enhancing existing QoS routing algorithms. Finally, an IVN routing optimization system is built and implemented, and the performance of different algorithms is compared and analysed. The experimental results show that compared with the traditional Dijkstra and ECMP algorithms, the DBROA algorithm can effectively reduce the data forwarding delay and packet loss rate, improve the overall performance of IVN, and provide better QoS guarantees for IVN real-time data transmission.

Keywords: software defined network, in-vehicle network architecture, network calculus, route optimization, DBROA algorithm.

1 Introduction

Currently, 5G is deployed around the world, and 6G has gradually entered people's vision. Automobiles in 6G life may become indispensable devices in addition to smartphones. Automobiles are accelerating the transformation to intelligent. Active safety, intelligent driving and intelligent Internet and other emerging businesses have promoted automobiles from a single transportation tool to a complex distributed computing system [9]. With the addition of new business, the number and complexity of onboard electronic equipment have surged, and the data in vehicles have exploded. The transmission, exchange and sharing of data make the in-vehicle network (IVN) system face serious challenges and bottlenecks in its development. The Renault-Nissan-Mitsubishi alliance developed a modular and real-time computing platform to cope with the technical challenges brought by future interconnected vehicles [7]. Kim et al. [19] designed an access gateway to realize data flow transmission between in-vehicle and out-vehicle networks. Zuo et al. [45] designed a CAN/CAN gateway to realize the conversion from the CAN/CANFD to the SOME/IP protocol. Lee et al. [23] designed the gateway between FlexRay and Ethernet protocol conversion and emphasized the importance of the Ethernet backbone in future vehicle systems. Walrand et al. [40] proposed an Ethernet based architecture to meet the requirements of vehicle communication bandwidth and simplify network configuration. Häckel et al. [11] systematically studied different strategies for integrating vehicular control flow with Ethernet network and analysed their security impact on software defined IVN. Rumez et al. [36] proposed a service oriented IVN architecture design and compared it with the signal oriented method.

The current IVN architecture presents the coexistence of multiple buses. The existing distributed computing systems are independent of each other, and there is little interoperability between networks. The design of an internal interconnection protocol stack is complex, and the scalability and openness are limited. There is a lack of unified network architecture design.

In view of the problems of electrical/electronic architecture (EEA), such as low flexibility, difficult expansion, poor compatibility, and its closed standards, it is not easy to connect with external equipment. Bandur et al. [1] discussed in detail the shortcomings of the traditional EEA and introduced the technologies needed to support new centralized architectures. Bucher [5] proposed an integration method of EEA development based on model and simulation, which effectively analysed and evaluated the bidirectional dependencies between architecture and behaviour. Huawei proposed a distributed network + domain controller CC architecture, which divides vehicles into three parts: driving, cockpit and vehicle control [16]. BYD launched the latest platform to develop the electronic domain control module of vehicle bodies to achieve multi-integration [42]. Li [25] proposed a design method for vehicle electronic and electrical communication networks based on SOME/IP. Bhatia et al. [2] proposed software definitions of vehicle development, vehicle information structure and technical systems. Nanjing Iveco realized the software upgrading of vehicle electronic control units through vehicle intelligent terminal controllers and self-built cloud platforms [37].

EEA is developing in a centralized direction. The demand for communication between different computing units and different domains in the vehicle is becoming increasingly stronger. The updating and addition of corresponding software functions require a more flexible architecture [12, 28, 41]. Therefore, the design of a unified management, flexible configuration of IVN architecture is particularly important for the current development of IVNs.

With the substantial increase in the functional requirements of smart vehicles, the number and complexity of software have rapidly increased. Different applications have different delay and quality of service (QoS) constraints on data transmission, which puts forward higher requirements for in-vehicle communication [6, 8, 14, 31, 34]. To meet a series of complex deterministic communication requirements, such as high bandwidth, low delay, low jitter and zero congestion in IVNs, IVNs need to provide fast and effective data transmission and limited delay for real-time operation. Ensuring the transmission quality of delay sensitive data brings new challenges to IVN technology [27]. Yang et al. [43] mentioned and analysed the end-to-end delay problem under the hybrid network architecture but did not give a specific optimization scheme. Kobayashi and Ito [21] studied the impact of time synchronization accuracy on time-aware shaping QoS. Lv et al. [29] formally modelled and analysed the time sensitive networking (TSN) scheduler using the time model checker UPPAAL, taking into account the transmission delay and time utilization of the TSN scheduler. Leonardi et al. [24] proposed a network partition system to reduce the impact of new flow entry on existing flow QoS in IVN. Kostić et al. [22] proposed a context-aware mobile network QoS prediction module for vehicular applications, using prediction parameters to optimize the operation of vehicular information entertainment systems to ensure better user experience. Zhou et al. [44] evaluated the implementation of multiple traffic scheduling and shaping mechanisms in automotive Ethernet and tested the performance of time-aware and asynchronous traffic shaping. Kim et al. [20] proposed a heuristic based Ethernet scheduling algorithm based on the Ethernet traffic scheduling standard of TSN technology. Syed et al. [38, 39] studied four different heuristic algorithms to solve scheduling and routing problems when the dynamic application program ran in the regional architecture. In addition, they also proposed the FTBH algorithm to dynamically schedule and route the key data of security. Nitta et al. [32] simulated and evaluated the QoS of the next generation vehicular Ethernet network for the strict priority queue (SPQ), reliable frame replication and elimination (FRER) and frame preemption (FP) mechanisms. Marino et al. [30] introduced a new network optimization queue strategy, the elastic queue engine (EQE), to maximize the QoS of time-sensitive data streams.

An effective routing optimization algorithm is the key factor to ensure QoS. In future vehicle development, it not only needs to deal with a large amount of indispensable data generated by its on-board sensors but also uses data from other vehicles and roadside units. Due to the existence of heterogeneous networks in the vehicle, the existing IVN routing technology cannot fully measure the data flow in the network and the relationship between business attributes and QoS parameters [35]. To overcome the drawbacks of the existing IVN architecture, improve the flexibility and scalability of IVNs, and ensure the low latency and highly reliable transmission of different types of data, this paper mainly makes the following contributions.

1. A new IVN architecture based on SDN is proposed. Based on SDN technology, a new IVN architecture for unified management and flexible configuration is designed to integrate additional hardware and applications and improve interoperability between heterogeneous network data sources. SDN introduces a centralized controller that manages and configures network devices through software, enhancing network flexibility. Administrators can dynamically configure the entire network via the SDN controller to meet the demands of various applications and services. This flexibility enables the network to adapt more quickly to changes, thereby improving service quality. SDN adopts a separated architecture with control and data planes, where the control plane is responsible for network management, and the data plane handles data transmission. This separation makes SDN easily scalable, allowing for the effortless addition of new network devices or nodes without altering the control plane. This enhances network scalability, making it suitable for large-scale deployments.

2. A QoS routing optimization mechanism based on the new IVN architecture is established. The performance of the multipriority scheduling strategy is analysed by introducing network calculus theory, and the end-to-end maximum delay model of IVN is established. Then, the model is introduced into the routing algorithm, the traditional routing algorithm is improved, and the DBROA algorithm is proposed. By comprehensively measuring the network attributes and QoS parameters, the dynamic selection of the optimal routing is realized. Finally, the simulation environment is built to verify the algorithm, and the performance is compared with the traditional Dijkstra and ECMP routing algorithms.

This paper is arranged as follows: Section 1 introduces the specific design of the new overall architecture of automobiles and the new IVN architecture based on SDNs. Section 2 summarizes the basic theory and formula derivation process of network calculus and establishes the end-to-end maximum delay model of IVN. Section 3 designs the IVN QoS routing optimization system and algorithm. In Section 4, the performance of the three algorithms is experimentally compared and analysed. The conclusion is summarized in the last part of this paper.

2 Design of the SDN based in-vehicle network architecture

With the application of advanced driving assistance systems (ADAS), vehicles to everything (V2X) and other technologies, vehicles have become more intelligent. To meet the development needs, the new overall architecture of open automobiles is used, as shown in Figure 1. The new overall architecture of automobiles uses Ethernet as the backbone network. Through the SDN central controller computing platform, it can realize the fusion of various information, realize vehicle coordination, online upgrade of software, and portability of the operating system, and enhance the interaction of IVN.

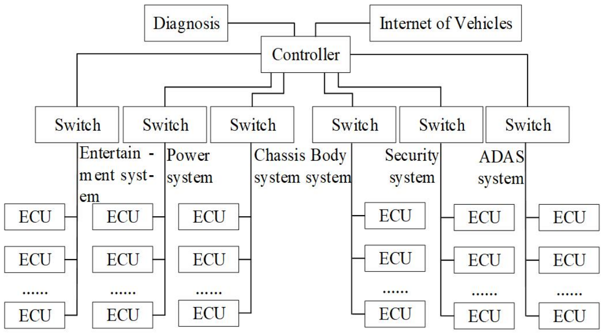

Figure 1: The new overall architecture of car

3 New in-vehicle network architecture design

The SDN is designed to standardize the control and configuration of the network infrastructure, which consolidates network control by porting the network control plane to a logical centralized controller and downgrading switches to simple forwarding devices. This provides great advantages for the development of IVNs in the future, including access control or reconfiguration based on the current network state centralized view, and provides a reliable method for interoperability between electronic control units. As shown in Figure 2, the new IVN architecture based on SDN aims to establish a real central processing integration with a vehicle domain controller as an integrated unit. It can overcome the shortcomings of the current in-vehicle communication architecture, such as weak interactivity and flexibility, and realize the full centralized intelligent control of the vehicle domain, which is more convenient for service and software expansion and upgrading.

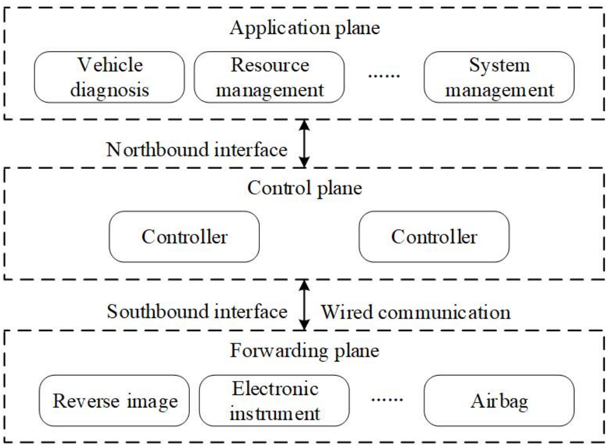

Figure 2: The SDN based new IVN architecture

The new IVN architecture is divided into three planes, which are the forwarding plane, the control plane and the application plane from bottom to top.

3.1 Forwarding plane

It is composed of sensors, actuators, OpenFlow switches and other hardware connected through physical links, corresponding to the SDN architecture. The flow table needed in the forwarding process is generated by the control plane, which connects the automobile parts with the application and management system to ensure the effective transmission of data. The OpenFlow switch contains multiple flow tables and group tables, and the internal structure diagram is shown in Figure 3. The flow on the data plane is divided into hard real-time flow, soft real-time flow, effort flow and network configuration flow. Hard real-time traffic comes from key security components, such as the power system/chassis control flow. All hard real-time traffic needs to be transmitted in time and must always be transmitted within a fixed delay. Soft real-time traffic is associated with less critical systems. If the delay requirement is not met, these systems may operate in a downgrade state, such as vehicle control flow and try to relate traffic to unsafe systems, such as entertainment. Network configuration traffic comes from various signalling messages between controllers and switches. In the scheduling program, priority queues are set for the data flow in the data plane to ensure the effective transmission of shared data in the vehicle. The new IVN improves the security performance of IVN using AUTOSAR SecOC and other systems to verify traffic and further defends invaders [14, 17, 33].

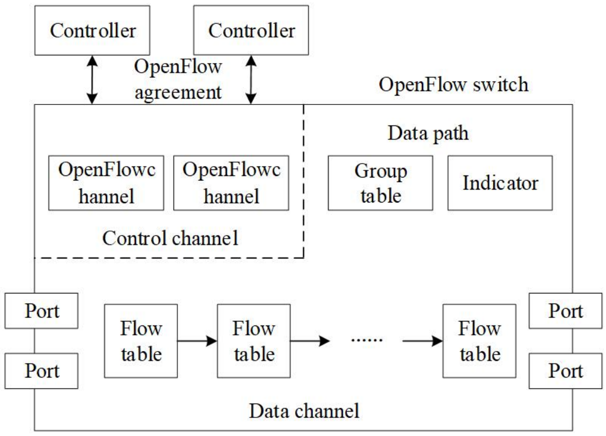

Figure 3: OpenFlow switch internal structure diagram

3.2 Control plane

The control centre of the system, SDN technology, integrates the control platform originally integrated in the hardware equipment into the SDN controller, and the SDN controller configures the forwarding plane through in-band/out-of-band communication. The control plane is mainly responsible for stream table sending, topology management and other functions. The network controller is directly connected to one of the switches, and the information from switches in IVN is monitored and collected at any time through the southward interface to record various logical networking information. The network controller controls the access to the northbound interface by reading the control list and different privilege levels. It uses the information from components (such as sensors and electronic control units) and applications to configure the network and timely feedback the link connection state of the network and the working state of the equipment.

3.3 Application plane

Without changing the underlying physical equipment, various services and applications are realized, and third parties or users are allowed to freely choose applications. These applications transmit the required network behaviour requests to the controller in a programmable manner. Simple network programming can realize the logic control of the network and the rapid online deployment of network applications. Applications are deployed only in the form of software programs, which can replace some hardware only functions in the traditional network. The global network view is obtained through the controller's northbound interface to reconfigure the network and manage the network resources effectively to ensure that all communications between automotive components and applications can proceed smoothly. The management system is responsible for authenticating components and applications, configuring and managing IVNs, and keeping lists of components and applications and their permissions. The external database contains information that may be added to the car, and the internal database is supplemented by the external database.

4 In-vehicle network delay model based on network calculus

Packet delay or loss in an IVN will have a great impact on driving stability and safety, and data flow must meet the delay guarantee required by real-time data flow in the transmission process. Therefore, this section mainly analyses the delay in IVN using network calculus theory, deduces the upper bound of queuing and sending delay, and establishes the end-to-end maximum delay model of IVN.

4.1 Network calculus

Network calculus, including deterministic network calculus and random network calculus, is a network performance analysis tool based on min plus and max plus [4]. Because the delay sensitivity of data flow should be guaranteed in IVN, this study selects the deterministic network calculus suitable for the worst case performance analysis framework of the network as the theoretical support. The following is a theoretical introduction to network calculus [13].

Definition 1: Generalized additive function set

Assuming that the function is continuous and that there is a first-order derivative, the generalized increment function set is defined as

$$\begin{gathered} F = \{ f(t) |\, f(t) \geq 0, \forall e \leq t, f(e) \leq f(t), t \in [0, +\infty)\} \\ F_0 = \{ f(t) |\, f(t) \in F, f\,(0) = 0\} \end{gathered} \tag{1}$$

Definition 2: Minimum convolution operation

$$(f \otimes g)(t) = \begin{cases} \inf\limits_{0 \leq e \leq t} \{f(t-e) + g(e)\} & t \geq 0 \\ 0 & t < 0 \end{cases} \tag{2}$$

Definition 3 Minimum plus deconvolution operations

$$(f \oslash g)(t) = \sup_{0 \leq e}\{f(t+e) - g(e)\} \quad \forall f, g \in F \tag{3}$$

Definition 4 Arrival curves

For any $\alpha(t) \in F$, if the data flow input cumulative function $A(t)$ satisfies $A(t) - A(e) \leq \alpha(t-e)$ for $0 \leq e \leq t$ at any time, the arrival curve $\alpha(t)$ is called $A(t)$, which can be expressed as

$$A(t) \leq (A \otimes \alpha)(t) \tag{4}$$

Definition 5 Service curve

Given an input cumulative function of system S and data flow $A(t)$, the output function of the system is $A^*(t)$, and if and only if function $\beta(t) \in F$ satisfies

$$A^*(t) \geq (A \otimes \beta)(t) \tag{5}$$

A service curve provided by a network node to data stream $A(t)$ is called $\beta(t)$

Definition 6 Upper limit of backlog

If the data flow with $\alpha(t)$ as the arrival curve passes through a system with $\beta(t)$ service curve, the flow backlog in the system satisfies

$$Q \leq \sup_{t \geq 0}\{\alpha(t) - \beta(t)\} \tag{6}$$

Definition 7 Delay upper bound

If the data flow with $\alpha(t)$ as the arrival curve passes through a system with $\beta(t)$ service curve, the data flow delay at the node satisfies

$$D \leq \inf_{t \geq 0}\{e \geq 0 : \alpha\,(t) \leq \beta\,(t+e)\} \tag{7}$$

4.2 Performance model based on network calculus

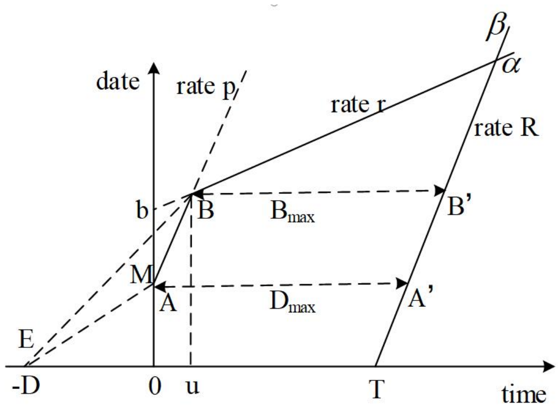

Figure 4: Basic performance model parameters

Usually, the arrival curve and service curve can be applied to network analysis to obtain the backlog and delay boundary of traffic in the system. The basic performance model parameters are shown in Figure 4 [33]. The arrival curve α is a token bucket flow that allows the flow to send up to b byte bursts, and the sustainable rate is $r \ \ Byte/s$, which can be expressed as $\alpha(t) = rt + b$. However, it is limited by link capacity C and maximum packet size M; that is, the arrival curve in the network can be expressed as

$$\alpha(t) = \min\{Ct + M, rt + b\} \tag{8}$$

Service curve $\beta(t)$ is as follows: before serving at the rate of $R \ \ Byte/s$, the data flow needs to wait T seconds, which can be expressed as

$$\beta(t) = R \cdot (t - T)^+ \tag{9}$$

where $(t - T)^+ = \max\{0, t - T\}$.

According to Formulas 6 and 7, the maximum distance between α and β in the vertical direction is

$$Q_{\max} = \max[\alpha(T), \alpha(u) - \beta(u)] \tag{10}$$

Among them,$u = \frac{b-M}{p-r}$ Similarly, according to the linear characteristics, line segments AA' and BB' are equal, and both are the maximum distances in the horizontal direction, namely, the upper bound of delay is

$$D_{\max} = \max(\frac{\alpha(t)}{R} + T - u, \frac{M}{R} + T) \tag{11}$$

Through the minimum additive algebra calculation, the backlog and delay bounds are

$$Q_{\text{inax}} = b + (\frac{b - M}{p - r} - T)^+[(p - R)^+ - p + r] + rT \tag{12}$$

$$D_{\max} = \frac{M}{R} + \frac{(b - M)(p - R)^+}{R(p - r)} + T \tag{13}$$

This paper assumes that the nonpreemptive strict priority scheduler is used when the token bucket flows through n queues, and the service curve of priority queue i is

$$\beta_i(t) = (Ct - t\sum_{j=1}^{i-1} r_j - \sum_{j=1}^{i-1} b_j - \max_{i+1 \le j \le n}\{p_j^{\max}\} - p_i^{\max})^+ \tag{14}$$

C is the link capacity, and r_j,b_j and p_j^{max} are the rate, burst size and maximum packet size of the token bucket flow through the queue, respectively. Corresponding to Equation 9, service rate R and waiting delay T_i can be expressed as

$$R_i = C - \sum_{j=1}^{i-1} r_j \tag{15}$$

$$T_i = \frac{\sum_{j=1}^{i-1} b_j + \max_{i+1 \le j \le n} \{p_j^{\max}\} + p_i^{\max}}{C - \sum_{j=1}^{i-1} r_j} \tag{16}$$

The corresponding backlog and delay bounds are expressed as

$$Q_i = b_i + r_j T_i \tag{17}$$

$$D_i = \frac{\sum_{j=1}^{i} b_j + \max_{i+1 \le j \le n} \{p_j^{\max}\} + p_i^{\max}}{C - \sum_{j=1}^{i-1} r_j} \tag{18}$$

4.3 End-to-end maximum delay model for vehicular networks

G and V are defined as physical and logical network graphs, respectively. $\mathcal{C}_{(s,w)}$ is the capacity of physical link$(s, w) \in G$. $E_{(s,w,p)} \in V$ represents a link queue with priority p from node s to node w, and the smaller is the value of p, the higher is the priority. Assuming that a set of data streams passed by the queue link is f, p_f^{max} is the maximum packet size of data stream f, and if unknown, it is represented by the maximum Ethernet frame p^{MAX}. The rate of stream f is r_f, and the burst of stream f is b_f. $F_{(s,w,p)}$ is the set of traffic flows through the queue link, and $F_{(s,w,p)}^S$ represents the sum of the rate of aggregation flow $F_{(s,w,p)}^S$, that is, $F_{(s,w,p)}^s = \sum_{f_{(s,w,p)}} r_f$. $F_{(s,w,p)}^J$ represents the sum of bursts of aggregation flow $F_{(s,w,p)}$, that is, $F_{(s,w,p)}^J = \sum_{f_{(s,w,p)}} r_f$. $F_{(s,w,p)}^{p^{Max}}$ denotes the maximum packet size in the aggregation flow, namely, $F_{(s,w,p)}^{p^{Max}} = Max_{f \in F_{(s,w,p)}} \{p_f^{Max}\}$. The buffer size of queue p is $E_{(s,w,p)}^J$, and the capacity size of queue p is $E_{(s,w,p)}^R$, $Max_{(s,w,p)}^T$ indicating the maximum burst allowed by the queue.

According to Formulas 17 and 18, the backlog limit and delay limit of the queue link can be expressed as

$$Q_{(s,w,p)} = F_{(s,w,p)}^J + F_{(s,w,p)}^S \frac{\sum_{j=1}^{p-1} F_{(s,w,j)}^J + \max_{p+1 \le j \le n} \{F_{(s,w,j)}^{p^{Max}}\} + F_{(s,w,p)}^{p^{Max}}}{C_{(s,w)} - \sum_{j=1}^{p-1} F_{(s,w,j)}^S} \tag{19}$$

$$D_{(s,w,p)} = \frac{\sum_{j=1}^{p} F_{(s,w,j)}^J + \max_{p+1 \le j \le n} \{F_{(s,w,j)}^{p^{Max}}\} + F_{(s,w,p)}^{p^{Max}}}{C_{(s,w)} - \sum_{j=1}^{p-1} F_{(s,w,j)}^S} \tag{20}$$

1) The packet size of the data flow cannot be larger than the maximum packet size in the network, namely,

$$F_{(s,w,p)}^{p^{Max}} \le p^{MAX} \tag{21}$$

2) The aggregate flow rate cannot exceed the capacity allocated for each queue by the resource allocation algorithm, namely,

$$F^{S}_{(s,w,p)} \leq E^{R}_{(s,w,p)} \tag{22}$$

3) The backlog of aggregated flows cannot exceed the buffer allocated to each queue by the resource allocation algorithm, namely,

$$Q_{(s,w,p)} \leq E^{J}_{(s,w,p)} \tag{23}$$

4) The burst of aggregated flows cannot exceed the maximum burst allowed by the queue, namely,

$$F^{J}_{(s,w,p)} \leq Max^{T}_{(s,w,p)} \tag{24}$$

Based on Formulas 21, 22 and 23, Formulas 19 and 20 can be rewritten as

$$Q_{(s,w,p)} \leq F^{J}_{(s,w,p)} + E^{R}_{(s,w,p)} \frac{\sum_{j=1}^{p-1} F^{J}_{(s,w,j)} + 2P^{MAX}}{C_{(s,w)} - \sum_{j=1}^{p-1} E^{R}_{(s,w,j)}} \leq E^{J}_{(s,w,p)} \tag{25}$$

$$D_{(s,w,p)} \leq \frac{\sum_{j=1}^{p} F^{J}_{(s,w,j)} + 2P^{MAX}}{C_{(s,w)} - \sum_{j=1}^{p-1} E^{R}_{(s,w,j)}} \tag{26}$$

Input Formula 24 into Formula 25 as follows

$$Max^{T}_{(s,w,p)} + E^{R}_{(s,w,p)} \frac{\sum_{j=1}^{p-1} Max^{T}_{(s,w,p)} + 2P^{MAX}}{C_{(s,w)} - \sum_{j=1}^{p-1} E^{R}_{(s,w,j)}} \leq E^{J}_{(s,w,p)} \tag{27}$$

$C_{(s,w)}$ is the link capacity, and P^{MAX} is the maximum packet size in the network, which can be represented by the maximum Ethernet frame. The queue capacity $E^{R}_{(s,w,p)}$ and the queue buffer $E^{J}_{(s,w,p)}$ are all allocated by the resource allocation algorithm. Therefore, Formula 27 is known data except that the queue allows maximum burst $Max^{T}_{(s,w,p)}$, and the value of $Max^{T}_{(s,w,p)}$ can be solved recursively

$$D_{(s,w,p)} \leq \frac{\sum_{j=1}^{p} Max^{T}_{(s,w,p)} + 2P^{MAX}}{C_{(s,w)} - \sum_{j=1}^{p-1} E^{R}_{(s,w,j)}} \tag{28}$$

$Max^{T}_{(s,w,p)}$ is obtained from Formula 27; its variables are all known data; thus, the upper bound of queue delay can be calculated by Formula 28. Formulas 19 and 20 are rewritten by scaling and replacing, and the queue delay independent of the network state is obtained.

By parsing the timestamp information encapsulated in LLDP packets, the time difference information of LLDP packets passing through the links between switches can be obtained [10]. LLDP scheduling refers to a method of managing and scheduling network devices using LLDP (Link Layer Discovery Protocol). LLDP allows network devices to broadcast their identification, configuration, and capability information in the network, enabling other devices to learn about their presence and characteristics. Combining the round-trip time difference between the controller and the switch calculated by the Echo message, the transmission delay between the switch links can be calculated. The LLDP and Echo packets are transmitted as shown in Figure 5 below.

The link delay T^{l} between switches 1 and 2 can be calculated as

$$T^{l} = \frac{T^{lldp}_{1} + T^{lldp}_{2} - T^{echo}_{1} - T^{echo}_{2}}{2} \tag{29}$$

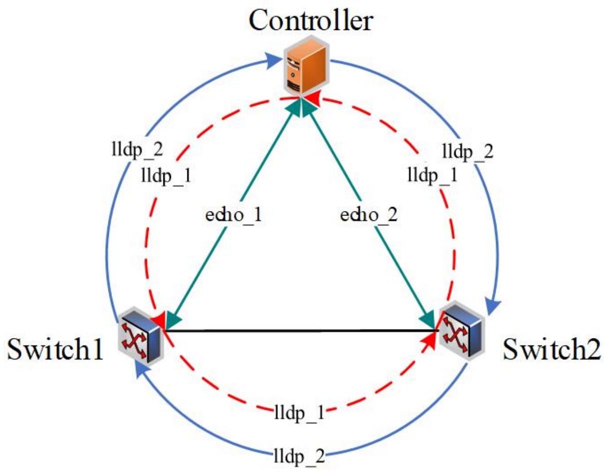

Figure 5: Measuring link delay

In IVN, the link delay formula is expressed as

$$T^{l}_{(s,w)} = \frac{T^{lldp}_{(s,w)} + T^{lldp}_{(s,w)} - T^{echo}_{s} - T^{echo}_{v}}{2} \tag{30}$$

Therefore, the maximum total end-to-end delay of the new IVN architecture based on SDN is

$$D_{ete} = \sum_{s=u}^{w=d} [D_{(s,w,p)} + T^{l}_{(s,w)}] + T^{echo}_{u} \tag{31}$$

where u is the source node, d is the destination node, and w is the next hop of node s.

4.4 Chapter summary

This chapter primarily focuses on determining and analyzing the key functional parameters affecting the deterministic behavior of real-time In-Vehicle Networks (IVN) communication using network calculus theory. It establishes an end-to-end maximum delay model based on network calculus in the context of a new architecture. Firstly, a detailed introduction to the Delay Network Calculus (DNC) and Server Network Calculus (SNC) theories in network calculus is provided, along with an explanation of fundamental mathematical definitions and reasoning within network calculus. Secondly, an analysis and derivation of the essential performance aspects of network calculus are presented, resulting in the constraint conditions necessary for computing complete scheduling schemes. Furthermore, upper bounds on latency and backlog are derived based on the Low-Latency Queue (LLQ) scheduling. Additionally, detailed insights into link delay measurements within the new architecture are provided. Finally, a comprehensive analysis of end-to-end delay components in the novel IVN architecture based on Software-Defined Networking (SDN) is conducted. This analysis leads to the establishment of an end-to-end maximum delay model under the new architecture, providing a theoretical foundation for end-to-end delay optimization in SDN-based IVNs. It also serves as a fundamental building block for subsequent dynamic routing optimizations, aiming to achieve deterministic and low-latency real-time IVN communication.

5 Design of QoS Routing Optimization System for IVN

5.1 System design

The IVN routing optimization system is mainly composed of the controller and the underlying forwarding equipment. By deploying the relevant detection module and routing algorithm in the controller, dynamic routing optimization based on the whole network structure and state is realized, and the overall performance of the IVN is further improved to ensure the transmission of real-time data. The relevant routing architecture design is shown in Figure 6.

The controller mainly deploys six modules: the receiving module, network sensing, network monitoring, information module, resource allocation and routing configuration.

When an IVN stream with QoS requirements arrives, the message information of the stream is sent to the controller. The controller receiving module sends the received packet information to the information module. The information module mainly records the information collected by each function module so that each module can be called at any time. The network perception module mainly monitors the change in network topology, updates the topology in time and obtains network topology information, switch port information and host information.

Network monitoring is used to obtain and analyse the state of the network in real time. Data, such as the flow velocity of the business flow, the flow table item, the port performance of the switch and the link performance, are counted. Through the collection and calculation of these parameters, the remaining bandwidth, link delay and packet loss rate can be obtained, and the relevant information is sent to the information module.

The resource allocation module dynamically adjusts the resources according to the network state information recorded in the information module and meets the requirements of all flows as much as possible from a global perspective. Additionally, the allocated information is stored in the information module. If network resources cannot withstand all flows, the priority of exit flows will be considered.

The routing configuration module is mainly based on the network monitoring and resource allocation module recorded by the information statistics module, according to the source and destination address through the global view to obtain path groups. By running the routing optimization algorithm, the optimal transmission path is calculated, and the optimal road runoff table is sent to the switch along the way through the flow table generated by the controller.

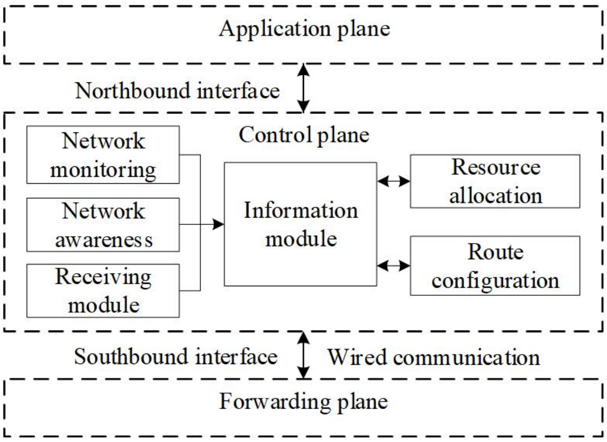

Figure 6: Frame diagram of routing optimization system

5.2 Routing arithmetic design

When packets of the new data stream arrive at the routing node, the routing reports to the controller, and the controller identifies the type of data stream. The Widest Shortest Path (WSP) algorithm, where 'W' represents available link bandwidth and 'S' represents the minimum number of hops, aims to select the shortest path with the highest remaining bandwidth. The primary emphasis of the WSP algorithm is on reducing network costs by choosing the path with the fewest hops, thus saving resource expenses, but this may increase the risk of network congestion. In this section, based on the WSP algorithm [18], the end-to-end delay model is combined to improve it, and a delay based route optimization allocation (DBROA) algorithm is designed. The DBROA algorithm mainly divides traffic into real-time traffic and nonrealtime traffic and uses different routing schemes to select the best routing path. The pseudocode of the DBROA algorithm is shown in Algorithm 1.

The specific steps of the DBROA algorithm are as follows. First, the QoS information of the network state and flow is obtained by the controller, and all routing paths are counted from the global view according to the source node and destination node information. If the number of paths is greater than 1, the controller determines the type of data flow; if the traffic is not real-time, the WSP algorithm is used to calculate the routing path. Otherwise, the maximum end-to-end delay is calculated according to the relevant information of the flow to determine whether it meets the IVN delay T^{QoS} requirements. If satisfied, the acceptable paths in the path group are judged according to the queue buffer $E^{J}_{(s,w,p)}$ of resource allocation and the bandwidth size $E^{D}_{(s,w,p)}$, and the number of paths satisfying the conditions N is recorded. If N=1, then return the path; if N>1, the shortest path is selected from the path satisfying the conditions as the optimal path; if the end-to-end maximum delay does not meet or there is no acceptable path in the path group, the path with the minimum end-to-end maximum delay is selected as the optimal path.

Algorithm 1 DBROA Algorithm

Input: network topology $G(V,E)$
Output: optimal path *path*

Get Path Group *paths* in G by Network Perception Module
if *paths.size()*>1 **then**
  if Flow is non-real-time flow **then**
    Select the widest and shortest path
  else
    Get the velocity of the flow r_f; Burst size b_f
    for all *path*∈*paths* **do**
      Calculation D_{ete} based on formula 31
      if $D_{ete} \leq T^{Qos}$ **then**
        if $b_f + F^{J}_{(s,w,j)} \leq E^{J}_{(s,w,p)}$ and $r_f + F^{S}_{(s,w,j)} \leq E^{D}_{(s,w,p)}$ **then**
          The paths satisfying the conditions are saved in *paths1* of the path group.
          if *paths.size()*=1 **then**
            This path is *path*
          else
            if *paths.size()*>1 **then**
              Select the shortest *path* from *path1* as a path
            end if
          end if
        else
          Select the smallest D_{ete} as *path*
        end if
      else
        Select the smallest D_{ete} as *path*
      end if
    end for
  end if
else if *paths.size()*=1 **then**
  Use this path as *path*
else
  path output null
end if
return *path*

6 Simulation Experiment

The environmental parameters in this experimental platform are shown in Table 1. In this paper, the Ryu controller and Mininet simulator are built on Ubuntu 16.04 to simulate the real IVN environment. The performance of the routing optimization algorithm under the new IVN architecture based on SDN is tested and compared with the traditional Dijkstra and ECMP algorithms [26].

Table 1: Experimental environment parameters

Experimental environment	Name	Version number
Windows10	Operating system	1903
VMware Workstation	Virtual machine	4.3.24
Ubuntu	Linux operating system	16.04LTS
Mininet	Network simulation tool	2.2.3.0d6
Ryu	Function Extension Controller	4.34

The experimental network topology is shown in Figure 7, which is mainly composed of one controller, five switches and eight hosts. This study uses the host to replace different ECUs for simulation experiments. Path1 represents path S1-S2-S3-S5, the link bandwidth is set to 8 Mbit/s, Path2 represents path S1-S4-S5, and the link bandwidth is set to 12 Mbit/s.

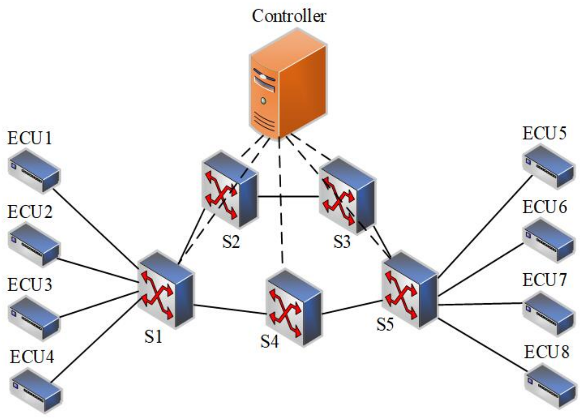

Figure 7: Experimental topological structure diagram

In this experiment, through the use of the Iperf tool and Ping command to test the performance of network links, Iperf can not only create TCP and UDP data streams with a variety of parameters and characteristics but also report the bandwidth, delay jitter, packet loss and other related information of network links [3]. The details of the four streams generated using Iperf are shown in Table 2.

Table 2: Sending data flow information

Type	Source, destination	Priority	Start time	Sending rate
Flow 1	ECU1- ECU5	1	0s	3 Mbps
Flow 2	ECU2- ECU6	2	5s	5 Mbps
Flow 3	ECU3- ECU7	3	10s	6 Mbps
Flow 4	ECU4- ECU8	1	15s	8 Mbps

In the network with limited resources, to verify that the proposed algorithm can achieve a more reasonable allocation of network resources according to the flow priority, this study uses the Wireshark tool to test the receiving rate of data flow from different sender ECUs. Figures 8, 9 and 10 show the test results of three different routing algorithms.

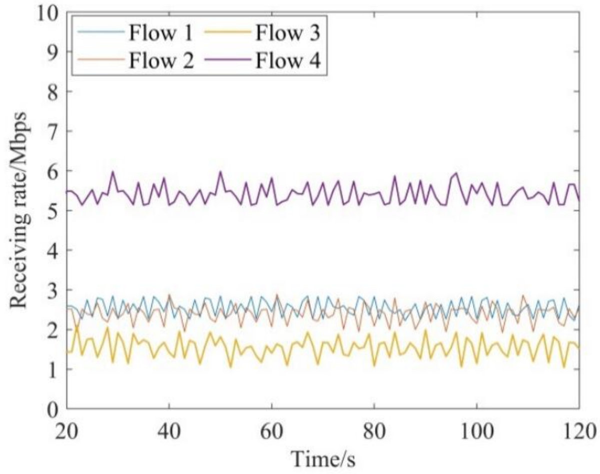

Figure 8: Receiving Rate of Dijkstra Algorithm

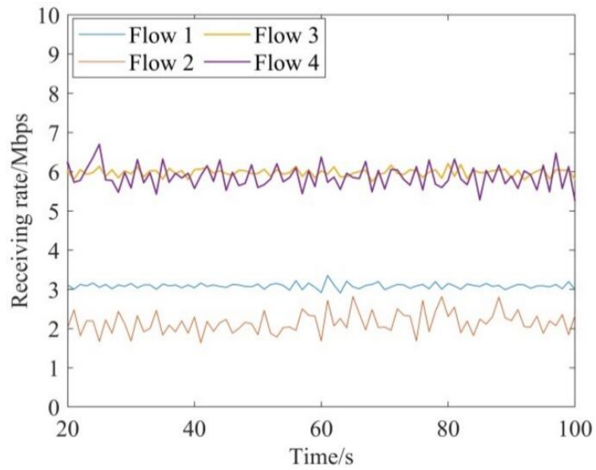

Figure 9: Receiving Rate of ECMP Algorithm

Figure 8 shows the test results of Dijkstra's algorithm. Due to the same source address and destination address of all traffic, a single path is used for traffic routing, resulting in congestion. Flow1 and Flow4 with high priority have poor performance. The transmission rate of Flow1 is maintained at approximately 2.5 Mbit/s, and that of Flow4 is maintained at approximately 5.4 Mbit/s. The receiving rate of data flow with high priority is less than the sending rate. In the Dijkstra algorithm, because low priority data streams occupy limited network resources, high priority data streams are not guaranteed by the QoS.

Figure 9 shows the test results of the ECMP algorithm. The flow is allocated to different paths through the equivalent allocation method. Flow1 and Flow3 use Path2, while Flow2 and Flow4 use Path1. This algorithm results in congestion in Path1, and the performance of Flow4 with high priority degrades. The transmission rate is approximately 5.7 Mbit/s, which is lower than the transmission rate. The ECMP algorithm cannot completely provide good service guarantees for the transmission of high priority rate streams.

Figure 10 shows the test results of the DBROA. It can be seen that data flow transmission with high priority is guaranteed. The receiving rates of Flow1, Flow2 and Flow4 are basically consistent with the sending rate. The receiving rate of Flow3 is approximately 3.5 Mbit/s, which is significantly lower than the sending rate. On the premise of meeting the end-to-end delay of a high priority data stream, this algorithm provides a QoS guarantee based on the priority of traffic. Compared with the other two algorithms, it better meets the QoS requirements of a high priority data stream.

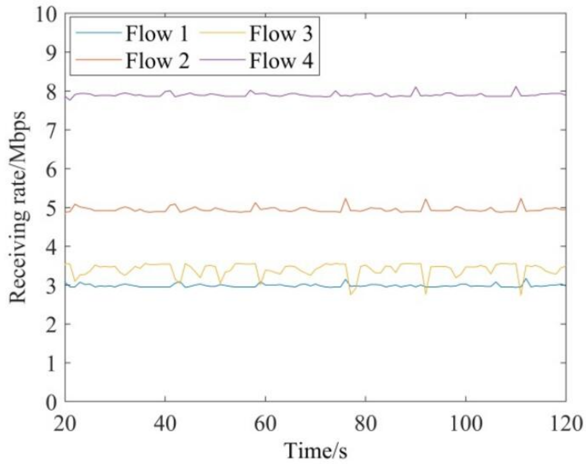

Figure 10: Receiving Rate of DBROA Algorithm

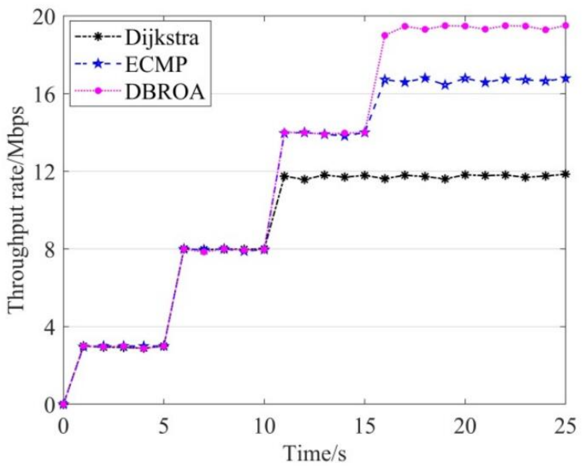

Figure 11: Throughput comparison

Figure 11 shows the comparison results of throughput of the three algorithms at different times. It can be seen that the throughput of the three algorithms is basically equal before 10s. For the Dijkstra algorithm, the three streams use the same path; thus, when t=10s, congestion will occur, the total throughput will not be higher than the link bandwidth, and the throughput tends to be approximately 11.7Mbps. For the ECMP algorithm, due to the existence of two paths, when the new flow arrives, it will be assigned to different paths. When t=10, two links will not have congestion. When t=15s, one of the links will have congestion. The total throughput is approximately 16.7Mbps, which is higher than that of Dijkstra's algorithm. The algorithm in this study fully considers factors such as delay, buffer size and link residual bandwidth from the global perspective. By optimizing the reasonable scheduling of the algorithm, the transmission of high priority flow is guaranteed, and the reasonable allocation of network resources is realized. The total link throughput reaches approximately 19.4Mbps, which is higher than that of the other two algorithms.

Different algorithms in the same network environment (when the network traffic is based on stability, P1, and P2 maximum bandwidth utilization) are shown in Figure 12. The utilization rate of the Dijkstra algorithm on the P1 path is 0, and the utilization rate on the P2 path is close to 100%, which is related to the fact that Dijkstra always chooses the path with the fewest hops. The ECMP algorithm selects a path for each data stream based on the load of the data stream. The utilization rate of P1 is close to 100%, and the utilization rate of the P2 path is approximately 75%. The algorithm does not make full use of the P2 residual band-width. The utilization ratio of the two links in this algorithm is close to 100%. This algorithm makes better use of network resources on the premise of avoiding congestion.

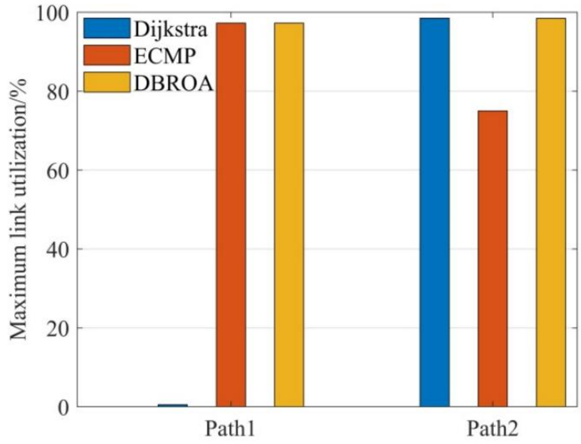

Figure 12: Maximum link bandwidth utilization

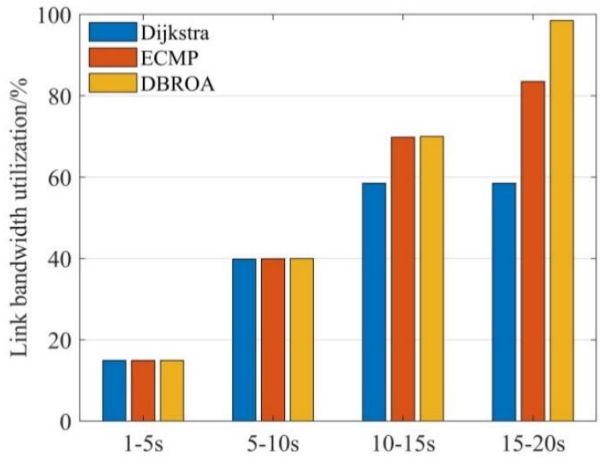

Figure 13: Total link bandwidth utilization

In addition, this paper compares the total link utilization of the three algorithms, and it can be seen from Figure 13 that the link utilization increases with increasing network traffic. Before 10 s, because all the links selected by the routing algorithm do not produce congestion, the overall utilization of the three algorithms is the same. When the network traffic is based on stability, it can be seen that the total link bandwidth utilization of this algorithm is higher than that of the other two algorithms.

Figure 14 shows the comparison results of the network packet loss rates of the three algorithms at different times. It can be seen that the performance of the three algorithms is similar due to the small sending rate of the sender before 10 s, and the network packet loss rate is small, almost 0. However, when the data flow is large, the Dijkstra algorithm forwarding all data streams according to the shortest path easily causes a shortage of link resources, resulting in network congestion. Therefore, the network packet loss rate rapidly increases and is high. The ECMP algorithm divides the data stream into path resources without considering the size of the flow, which easily causes local congestion in the network; thus, the packet loss rate is basically maintained at approximately 24% compared with Dijkstra's algorithm. The algorithm in this study considers the link state information and dynamically allocates high priority data streams to the optimal path for forwarding. The load balancing effect is good, and the packet loss rate is the lowest, approximately 11%.

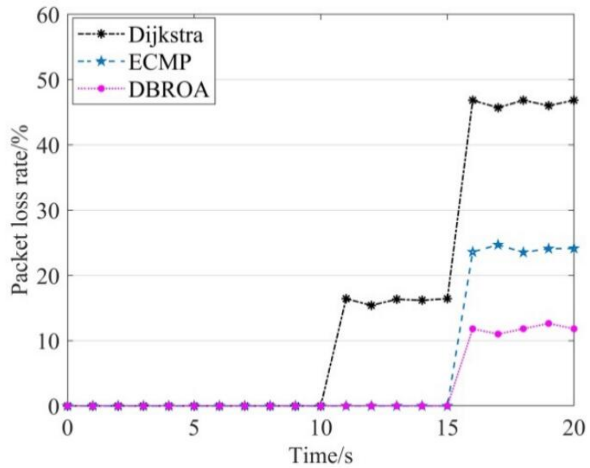

Figure 14: Comparison of average packet loss rate

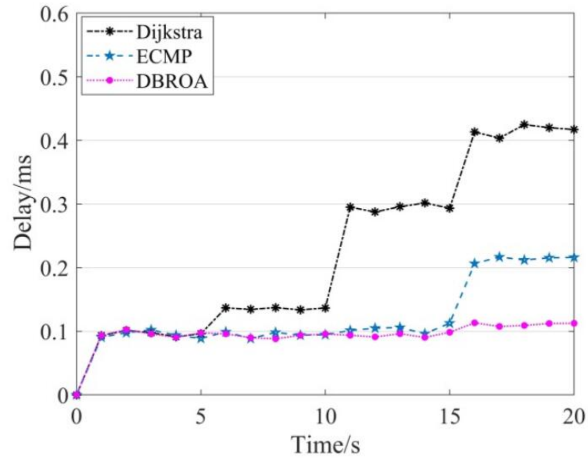

Figure 15: Comparison of average packet loss rate

Figure 15 shows the average end-to-end delay of the three algorithms for high priority data streams at different times. When the path load in the first 5 s is low, the end-to-end delay of the three algorithms is similar. In Dijkstra, with the entry of new streams, the path load is increased, and the available resources of the network are squeezed, resulting in a large increase in data flow delay after 5 s. Because the ECMP algorithm assigns new flows to different paths and achieves a certain degree of load balancing, the end-to-end delay changes little before 10 s. After 10 s, the end-to-end delay of high priority data streams increases, but the delay increase is less than that of Dijkstra's algorithm because the new stream enters the high priority flow link to occupy the available resources. Because the algorithm in this paper strictly guarantees the transmission of high priority flow and does not let the low priority flow occupy too much network resources of high priority flow, it can be seen that the end-to-end delay of the algorithm in this paper is basically stable before 15 s, and there is a small increase after 15 s. With the decrease of available network resources, this algorithm selects a more suitable path for data packets, which effectively reduces the end-to-end delay of high priority data streams compared with the other two algorithms.

7 Conclusion

Aiming to solve the problems of low interactivity and weak scalability of existing IVNs, this paper introduces SDN technology and designs a new IVN architecture based on SDNs. This architecture can effectively improve IVN flexibility and facilitate the integration and processing of vehicle data. At the same time, to meet the QoS requirements of vehicle applications, especially the real-time guarantee, the routing optimization problem under the new architecture is studied. Based on network calculus theory, the end-to-end maximum delay model of IVN is proposed, the traditional routing algorithm is improved, and the DBROA algorithm is designed. By building the QoS routing optimization system, the algorithm is verified by experiments. The experimental results show that, compared with the traditional Dijkstra and ECMP algorithms, this algorithm has better performance in throughput, end-to-end delay, and link bandwidth utilization, which effectively realizes the reasonable allocation of resources and provides QoS guarantee for the transmission of real-time traffic.

However, compared with the traditional routing optimization algorithm, the routing optimization algorithm in this study has higher computational complexity and is insufficient in running time. In future research work, the problem of dynamic traffic scheduling and resource allocation in the control plane will be deeply studied and analysed, and a routing algorithm with low computational complexity, short running time and excellent routing effect will be further designed to better optimize the network performance of IVNs. By leveraging relevant encryption algorithms and privacy protection authentication methods, we can enhance the security of In-Vehicle Networks (IVN) to safeguard the secure transmission of IVN data. Comparative analysis with other theoretical performance metrics can be introduced to assess strengths and weaknesses, and further theoretical frameworks can be incorporated to analyze latency performance comprehensively.

In the future, aligning with the functional requirements of connected and intelligent vehicles, the development of corresponding application software can be pursued. To better reflect the accuracy of optimization algorithms, real-world vehicle testing and validation would be essential.

Acknowledgements

This work were supported by the key scientific research program of Hubei Provincial Department of Education (D20211802) and Key R&D Plan Project of Hubei Provincial Department of Science and Technology (2022BEC008).

Author contributions

Yahong Zhai (b. March 19,1979) received her Master's degree in Computer Science Engineering (2007) from Wuhan University of Science and Technology. Now she is associate professor and master's supervisor of School of Electrical and Information Engineering, Hubei University of Automotive Technology. Her current research interests include different aspects of Computer Networks and Big Data Technology. She has(co-)authored more than 20 projects and 40 papers.

Yong Lv (b. November 14, 1994) received his Master's degree in Computer Science Engineering (2022) from Hubei University of Automotive Technology. His current research interests include different aspects of Computer Networks and Big Data Technology. He has(co-)authored more than 10 papers.

Longyan Xu (b. November 19, 1982) received his Master's degree in Automation (2006) from Hubei University of Automotive Technology. Now he is associate professor and master's supervisor of School of Electrical and Information Engineering, Hubei University of Automotive Technology. His current research interests include different aspects of Industrial Internet and Enterprise Automation. He has(co-)authored more than 20 projects and 30 papers.

References

[1] Bandur, V.; Selim, G.; Pantelic, V.; Lawford, M. (2021). Making the case for centralized automotive E/E architectures. *IEEE Transactions on Vehicular Technology*, 70(2): 1230-1245.

[2] Bhatia, J.; Modi, Y.; Tanwar, S.; Bhavsar, M. (2019). Software defined vehicular networks: A comprehensive review. *International Journal of Communication Systems*, 32(12): e4005.

[3] Bhardwaj, S.; Panda, S.N. (2022). Performance evaluation using RYU SDN controller in software-defined networking environment. *Wireless Personal Communications*, 122(1): 701-723.

[4] Bouillard, A. (2021). Individual service curves for bandwidth-sharing policies using network calculus. *IEEE Networking Letters*, 3(2): 80-83.

[5] Bucher, H. (2020). Integrierte modellund simulationsbasierte Entwicklung zur dynamischen Bewertung automobiler Elektrik/Elektronik-Architekturen. *Karlsruher Institut für Technologie (KIT).*

[6] Choi, E.; Song, H.; Kang, S.; Choi, J.W. (2021). High-speed, low-latency in-vehicle network based on the bus topology for autonomous vehicles: Automotive networking and applications. *IEEE Vehicular Technology Magazine*, 17(1): 74-84.

[7] David, R.; Li, S.; Dore, P. (2021). Modular and open platform for future automotive computing environment. *Multi Processor System on Chip 2: Applications*, 95.

[8] Deng, L.; Xie, G.; Liu, H.; Han, Y.; Li, R.; Li, K. (2022). A survey of real-time ethernet modeling and design methodologies: From AVB to TSN. *ACM Computing Surveys (CSUR)*, 55(2): 1-36.

[9] Guo, H.; Zhou, X.; Liu, J.; Zhang, Y. (2022). Vehicular intelligence in 6G: Networking, communications, and computing. *Vehicular Communications*, 33: 100399.

[10] Guo, X.; Wang, C.; Cao, L.; Jiang, Y.; Yan, Y. (2022). A novel security mechanism for software defined network based on Blockchain. *Computer Science and Information Systems*, 19(2): 523-545.

[11] Häckel, T.; Meyer, P.; Korf, F.; Schmidt, T.C. (2022). Secure time-sensitive software defined networking in vehicles. *IEEE Transactions on Vehicular Technology*, 72(1): 35-51.

[12] Huang, J.; Zhao, M.; Zhou, Y.; Xing, C.C. (2018). In-vehicle networking: Protocols, challenges, and solutions. *IEEE Network*, 33(1): 92-98.

[13] Huang, Z.Q.; Xu, B.J.; Zhang, Y.X.; Li, J.M. (2018). Software defined QoS routing algorithm for internet of vehicles based on network calculus. *Computer Applications*, 38(S2): 201-205.

[14] Jain, A.; Nandan, D.; Meduri, P. (2023). Data export and optimization technique in connected vehicle. *Ingénierie des Systèmes d'Information*, 28(2): 517-525.

[15] Jain, A.; Nandan, D.; Meduri, P. (2023). Data export and optimization technique in connected vehicle. *Ingénierie des Systèmes d'Information*, 28(2): 517-525.

[16] Jia, W.W.; Xu, K.Y.; Wang, H.B.; Deng, X.L.; Wang, Z.G. (2020). Smart car electronic architecture analysis and research. *Time Car*, 2020(4): 43-46.

[17] Jichici, C.; Groza, B.; Ragobete, R.; Murvay, P.S.; Andreica, T. (2022). Effective intrusion detection and prevention for the commercial vehicle SAE J1939 CAN bus. *IEEE Transactions on Intelligent Transportation Systems*, 23(10): 17425-17439.

[18] Ju, J.T.; Luo, Q.; Zheng, H.D. (2019). Routing control mechanism based on link characteristics in SDN. *Computer Applications*, 39(S1): 138-142.

[19] Kim, D.Y.; Jung, M.; Kim, S. (2020). An internet of vehicles (IoV) access gateway design considering the efficiency of the in-vehicle ethernet backbone. *Sensors*, 21(1): 98.

[20] Kim, H.J.; Choi, M.H.; Kim, M.H.; Lee, S. (2021). Development of an ethernet-based heuristic time-sensitive networking scheduling algorithm for real-time in-vehicle data transmission. *Electronics*, 10(2): 157.

[21] Kobayashi, C.; Ito, Y. (2021). Study on effect of time synchronization precision on QoS with time-aware shaper. In *2021 IEEE 10th Global Conference on Consumer Electronics (GCCE)*, Kyoto, Japan, pp. 639-640.

[22] Kostić, S.; Bjelica, M.; Tošić, N.; Kovačević, B. (2021). Prediction of mobile network QoS on the go for in-vehicle infotainment usage. In *2021 Zooming Innovation in Consumer Technologies Conference (ZINC)*, Novi Sad, Serbia, pp. 182-184.

[23] Lee, T.Y.; Lin, I.A.; Liao, R.H. (2020). Design of a FlexRay/Ethernet gateway and security mechanism for in-vehicle networks. *Sensors*, 20(3): 641.

[24] Leonardi, L.; Bello, L.L.; Patti, G. (2021). Bandwidth partitioning for Time-Sensitive Networking flows in automotive communications. *IEEE Communications Letters*, 25(10): 3258-3261.

[25] Li, Y.C. (2020). EE communication network design and research based on SOME/IP. *Automotive Digest*, 2020(8): 32-38.

[26] Liu, Z.P.; Zhang, Q.W.; Li, M.; Li, Z.Y.; Li, X.F. (2022). SDN routing strategy based on ant colony optimization algorithm. *Journal of Kunming University of Technology (Natural Science Edition)*, 47(3): 60-66.

[27] Lo Bello, L.; Patti, G.; Vasta, G. (2021). Assessments of real-time communications over TSN automotive networks. *Electronics*, 10(5): 556.

[28] Lu, S.; Shi, W. (2021). The emergence of vehicle computing. *IEEE Internet Computing*, 25(3): 18-22.

[29] Lv, J.; Zhao, Y.; Wu, X.; Li, Y.; Wang, Q. (2020). Formal analysis of TSN scheduler for real-time communications. *IEEE Transactions on Reliability*, 70(3): 1286-1294.

[30] Mariño, A.G.; Fons, F.; Gharba, A.; Ming, L.; Arostegui, J.M.M. (2021). Elastic queueing engine for time sensitive networking. In *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, Helsinki, Finland, pp. 1-7.

[31] Metwaly, S.S.; El-Haleem, A.M.A.; El-Ghandour, O. (2021). No-regret matching game algorithm for NOMA based UAV-Assisted NB-IoT systems. *Ingénierie des Systèmes d'Information*, 26(1): 79-85.

[32] Nitta, M.; Ito, Y.; Hayakawa, M. (2021). QoS evaluation of combination SPQ, FRER and FP for in-vehicle networks. *IEICE Communications Express*, 10(12): 894-898.

[33] Okokpujie, K., Mughole, D., Badejo, J.A., Adetiba, E. (2022). Congestion intrusion detection-based method for controller area network bus: A case for KIA SOUL vehicle. *Mathematical Modelling of Engineering Problems*, 9(5): 1298-1304.

[34] Pallavi, C.H.; Sreenivasulu, G. (2021). A high speed underwater wireless communication through a novel hybrid opto-acoustic modem using MIMO-OFDM. *Instrumentation Mesure Métrologie*, 20(5): 279-287.

[35] Park, J.S.; Kim, D.H.; Suh, I.H. (2021). Design and implementation of security function according to routing method in automotive gateway. *International Journal of Automotive Technology*, 22: 19-25.

[36] Rumez, M.; Grimm, D.; Kriesten, R.; Sax, E. (2020). An overview of automotive service-oriented architectures and implications for security countermeasures. *IEEE Access*, 8: 221852-221870.

[37] Song, W.; Hu, Q.S.; Tang, J.A. (2019). Over the air download technology application in commercial vehicles. *Auto Appliances*, 2019(12): 8-11.

[38] Syed, A.A.; Ayaz, S.; Leinmüller, T.; Chandra, M. (2021). Dynamic scheduling and routing for TSN based in-vehicle networks. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, Montreal, QC, Canada, pp. 1-6.

[39] Syed, A.A.; Ayaz, S.; Leinmüller, T.; Chandra, M. (2021). Fault-tolerant dynamic scheduling and routing for TSN based in-vehicle networks. In *2021 IEEE Vehicular Networking Conference (VNC)*, Ulm, Germany, pp. 72-75.

[40] Walrand, J.; Turner, M.; Myers, R. (2021). An architecture for in-vehicle networks. *IEEE Transactions on Vehicular Technology*, 70(7): 6335-6342.

[41] Xie, Y.; Zhou, Y.; Xu, J.; Zhou, J.; Chen, X.; Xiao, F. (2021). Cybersecurity protection on in-vehicle networks for distributed automotive cyber physical systems: State of the art and future challenges. *Software: Practice and Experience*, 51(11): 2108-2127.

[42] Yang, Y.P. (2019). Biyadi pure electric vehicle e platform technology analysis. *Automobile Maintenance and Repair*, 2019(23): 66-69.

[43] Yang, C.Y.; Ruan, H.T.; Zhang, Y.F.; Chen, K. (2019). Automotive general hybrid network architecture design. *Journal of Hubei Automotive Industry University*, 33(2): 5-10.

[44] Zhou, Z.; Lee, J.; Berger, M.S.; Park, S.; Yan, Y. (2021). Simulating TSN traffic scheduling and shaping for future automotive Ethernet. *Journal of Communications and Networks*, 23(1): 53-62.

[45] Zuo, Z.; Yang, S.; Ma, B.; Zou, B.; Cao, Y.; Li, Q.; Li, J. (2021). Design of a CANFD to SOME/IP gateway considering security for in-vehicle networks. *Sensors*, 21(23): 7917.

Cite this paper as:

Zhai, Y.H.; Lv, Y.; Xu, L.Y. (2024). Optimized QoS Routing in Software-Defined In-Vehicle Networks, *International Journal of Computers Communications & Control*, 19(1), 5962, 2024.
https://doi.org/10.15837/ijccc.2024.1.5962