IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

SPECIAL SECTION ON MOBILE SERVICE COMPUTING WITH INTERNET OF THINGS

Received January 21, 2019, accepted February 11, 2019, date of publication February 14, 2019, date of current version March 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2899402

A Queuing Delay Utilization Scheme for On-Path Service Aggregation in Services-Oriented Computing Networks

MINGFENG HUANG[1], WEI LIU[2], TIAN WANG[3], HOUBING SONG[4], XIONG LI[5], AND ANFENG LIU[1]

[1]School of Computer Science and Engineering, Central South University, Changsha 410083, China
[2]School of Informatics, Hunan University of Chinese Medicine, Changsha 410208, China
[3]College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China
[4]Department of Electrical, Computer, Software, and Systems Engineering, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA
[5]School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

Corresponding author: Anfeng Liu (afengliu@mail.csu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61572528, Grant 61772554, and Grant 6157256, and in part by the Central South University Postgraduate Independent Exploration and Innovation Project under Grant 2018zzts568.

ABSTRACT In services-oriented computing networks, packets in the process of routing to a data center must wait for a sufficient amount of data before service aggregation to reduce the network transmission load. However, packets must be uploaded to the data center as soon as possible to reduce delay. With the exponential growth in the number of IoT connected devices, the wait time for packets is longer at routers due to massive amounts of data, which causes a large queuing delay. If this queuing time can be utilized for service aggregation in a service-oriented computing network, the network performance will be substantially improved. Therefore, a queuing delay utilization scheme for on-path service aggregation (SAQD) is proposed in this paper. This scheme has the following innovations: 1) SAQD fully utilizes the queuing delay of packets for service aggregation, which can effectively reduce the transmission volume and communication overhead. Based on the proposed service aggregation algorithm, packets are divided into forwarding packets and aggregating packets, and the service aggregation of aggregating packets is completed by utilizing the transmission time of forwarding packets to ensure that the transmission volume and communication overhead are effectively reduced without additional latency. 2) SAQD can effectively alleviate the traffic pressure of the data center and balance the workload of routers. By the service aggregation and intranet cache of routers, some requests for the data center can be handled by routers, which reduces the traffic pressure of the data center, especially in the peak period. Compared with conventional schemes, the experimental results demonstrate that SAQD reduces the workload of the data center by 55.8%–66.26% and provides users with a better quality of experience by reducing the request response delay by 31.33%∼51.41%.

INDEX TERMS Internet of things, big data, queuing delay, service aggregation.

I. INTRODUCTION

Due to the rapid development of mobile devices [1]–[5] and wireless technologies [6], [7], a vast number of mobile devices, such as smart phones [4], [5], [8]–[11], wireless sensors [12]–[17], and devices in vehicles [5], [8], [18], [23], have an increasingly important role in our daily lives and are profoundly changing the current network structure and computing model [19]–[23]. The focus of network architecture shifts from the network center to the network edge [2], [8], [24]–[27]. Numerous devices connected to the Internet and various sensing devices considerably expand the ability of humans to perceive the world and acquire data [28], which enables the implementation of many applications that were previously unfeasible [29]–[31]. According to [32], since 2011, the number of objects connected to the Internet of Things (IoT) worldwide—9 billion—has exceeded the total global population. By 2020, the number of devices connected

The associate editor coordinating the review of this manuscript and approving it for publication was Wenbing Zhao.

to the Internet is estimated to be 24 billion [32]. Due to the combination of big data [33] and current artificial intelligence technologies such as deep learning [28], [34], many applications have been developed. For example, WeatherLah is an application that is based on meteorological data collection [30], and Waze is an application of real-time traffic information [31]. In addition, the computing power of these mobile devices has been qualitatively developed. With the vast potential of the computing resources of mobile devices on the edge of a network, applications and data begin to be placed on the network edge instead of being concentrated at the network center [2], [35]–[37]. Edge computing is the result of this shift in computing mode [3], [25], [38]. (2) Service computing is an emerging model for network computing and serves as a bridge among systems, man, and cybernetics. Service computing encompasses the science and technology of connecting the gap between business and IT services and has attracted an increasing amount of attention from both industry and academia [4], [25], [39], [40]. The number of mobile devices on the network edge is immense and their computing power is considerably greater than the computing power of personal computers 10 years ago due to the development of micro processing technology; thus, a substantial amount of computing can be conducted on the network edge [3], [7], [25]. Especially with the development of the Software Define Network (SDN) [5], [16], [18], [28], mobile devices have more functions by recompiling new programs, which have a broad application prospect [5], [16]. The combination of data, computing power and the SDN forms the material and technical foundation for the service computing mode [2], [4], [35], [41], [42]. What's more, the network based on data transmission encounters many difficulties, which accelerate the transformation of the service computing mode. A report from Cisco indicates that the data traffic generated by the IoT accounted for 69% of the total Internet traffic in 2014, which is 30 times the data traffic in 2000, and the data traffic is exponentially increasing [32]. The unprecedented growth of mobile devices and data traffic has caused a sharp increase in network congestion and delay, leading to a sharp deterioration in the Quality of Services (QoS) and a poor Quality of Experience (QoE), which causes the networks based on data transmission to face challenges [35]–[37]. Therefore, some researchers propose the Orchestrating Data as Service Networking (ODSN) framework [36]. In ODSN, the data center distributes software to network devices at different layers, and then devices orchestrate big data to services to implement a network model that is centered on service computing instead of data computing [36], [38].

According to [2] and [39], services of IoT are defined as software artifacts that are self-described, reusable, and highly portable, which are synthesized from collected data. These services are basic units that can form larger services or applications by integrating with services or data on other devices. Thus, mobile service computing enables us to provide and access services anytime and anywhere, which substantially facilitates our life, work, and studies. However, the application of mobile service computing has challenges. The main goal of service computing is to change the overload of the conventional data-based network caused by the transmission of raw sensing data [39], [40]. Service-oriented networks face similar challenges. Although service-oriented networks can reduce network traffic by orders of magnitude, mobile devices and emerging new applications are also growing at an order of magnitude. Therefore, ways to effectively reduce the load of network transmission and delay in routing remains a challenging issue. In previous studies, we proposed a service aggregation scheme that enables services with the same attributes to be transmitted along the same route and aggregated with other data over the route, which reduces network traffic [40]. However, this research does not consider additional delays caused by service aggregation. For routing in the services computing mode, it has a unique particularity. In the service computing mode, during the process of transmitting packets to a data center, devices at different levels are orchestrated into services with smaller capacities and stronger functions. Devices simultaneously cache services locally, which enables services to be satisfied locally when requested by users, which considerably reduces the latency of service requests. To minimize the data transmission load, devices wait as long as possible for sufficient packets to perform unified integration, which minimizes the traffic that needs to be transmitted. However, time is required to wait for more data and to perform service aggregation. Therefore, reducing the amount of data that is transmitted at the same time ensuring network latency is a key problem to be solved in service computing networks.

We note that mobile devices and data are exploding in the context of IoT and big data, and the data processing capabilities of network devices such as routers are limited. Therefore, the data load of routers is increasing, which causes the packets to remain in the queue for a longer period in the process of being transmitted to the data center, which forms a large queuing delay. The wait time of packets in the queue can be utilized for data processing and service aggregation. In this way, the lengthy queue delay is reasonably utilized, the extra time required for service aggregation is omitted, and the amount of data that is transferred is considerably reduced.

Based on this idea, a queuing delay utilization scheme for on-path service aggregation (SAQD) is proposed in this paper, which is especially suitable for services-oriented computing networks with intensive data. Compared with previous schemes, SAQD has the following innovations:

(1) SAQD fully utilizes the queuing delay of packets for service aggregation. When the queue length attains the aggregation threshold, packets are classified as aggregating packets and forwarding packets based on the service aggregation algorithm, and forwarding packets are sent to the next hop with the principle of First In First Out (FIFO). Aggregating packets are synthesized at the router as a reusable service by utilizing the data transmission time of forwarding packets. After the service aggregation is completed, aggregating

packets are converted into forwarding packets in the next round of aggregation and are sent to the next hop in accordance with the normal queue order, which enables effective use of queuing delay. For transmission delay and propagation delay, we propose a dynamic relay selection algorithm that can guarantee the total delay of packets.

(2) SAQD effectively reduces the transmission data amount and communication overhead by service aggregation. Each time a packet passes through a router, it will be aggregated with other packets received by the router. As the number of aggregations increases, the amount of transmitted data gradually decreases, which indirectly reduces the delay due to the amount of transmitted data that affects the transmission delay. Multiple packets are aggregated into a high-quality service that is cached locally at routers. When the stored service is requested by users, it can be directly returned from the router, which avoids lengthy network routing and reduces the transmission overhead of routers.

(3) SAQD substantially relieves the traffic pressure of the data center and balances the workload of routers. By orchestrating data packets into services and caching them in routers, some requests of the data center can be replaced by routers, especially when the traffic peaks, the traffic pressure of the data center can be substantially alleviated. Moreover, returning services from routers realizes the near access of data and improves the QoE of users. In addition, under the shortest queue priority relay selection algorithm, the relay selection of each hop is dynamic and flexible, which ensures that the data volume of routers is more balanced.

(4) The experimental results demonstrate that for the same network conditions, the SAQD scheme outperforms the conventional FSR scheme, which is a scheme that adopts the shortest distance route and the same queue scheduling method as in this paper without service aggregation. Compared with FSR, SAQD improves the request response delay by 31.33%∼51.41%, balances the data load of routers nearly doubled, and guarantees the delay.

The remainder of this paper is organized as follows: In Section II, a literature review is presented. The system model is presented in Section III. In Section IV, we introduce the SAQD scheme. The experimental results of SAQD are presented in Section V. Section VI provides the conclusions.

II. RELATED RESEARCH

In the context of services-oriented computing networks with big data, the main challenges faced by a network are (1) how to disperse the workload of the data center and balance the traffic load of routers. (2) How to respond to users' requests more quickly and provide users with a better Quality of Experience.

A. RESEARCH ON LATENCY

Latency is the time required to transfer packets to a data center, including propagation delay, transmission delay and queuing delay [43]. Propagation delay is determined by distance and propagation speed. Therefore, many researchers attempt to reduce the single-hop propagation delay using the shortest route; thus, the node closest to the sending node is prioritized when selecting the next relay. Although the shortest route algorithm can guarantee the age of information of the packets while minimizing the single hop delay, it will increase the total number of transmission hops, and each hop relay selection may require additional time. Conversely, packets are sent from the source node to the data center via multiple routers. The farther each router can transmit, the smaller is the total number of relay hops, and the end-to-end propagation delay may be reduced. Based on this idea, Papadopoulos *et al.* proposes a MobiDisc scheme in [44] to reduce the total propagation delay by maximizing the single-hop transmission distance.

Transmission delay is related to packet size and transmission bandwidth. Data fusion is the technology that is employed to reduce the amount of data. In all data collection-based networks, packets that are generated at the same time or in the same space tend to have a strong correlation, and data fusion can effectively reduce the amount of data transmitted by removing redundant data between multiple packets. In [45], Aparecido Villas et al. propose a local information fusion strategy that aggregates different packets in a small area onto a routing path, and packets are merged at each node on the path.

Data fusion is generally applied in sensing networks, while service aggregation is a more efficient way to reduce the amount of data that is transferred in cloud computing or fog computing systems. The concept of service aggregation was introduced by Ding *et al.* [39] in a paper on services networks. Service aggregation refers to the fusion of multiple packets, which are encapsulated into a service packet; the encapsulated service is directly returned when the data is requested without data reprocessing. In general, a service is attributed to data processing; thus, its capacity is considerably smaller than the original data.

Although increasing the transmission bandwidth is an effective way to reduce the transmission delay, it is limited by the network environment and space area, and the increase in bandwidth requires immense network costs.

Queuing delay refers to the time that packets remain in the forwarding queue. If a data packet is sent to an overloaded relay node, a large waiting delay is generated, and simultaneously, other idle nodes are not effectively utilized. In previous studies, a queue length threshold or timer is usually set to schedule the queue. When the queue length attained the threshold or the waiting time arrived, the data dequeues.

Li *et al.* [46] propose an AAR aggregation routing algorithm. In the routing policy, the sender dynamically selects the next node according to the length of its queue. Packets are preferentially assigned to the node with the longest queue length; thus, the queue can attain the length threshold as soon as possible to reduce the data waiting time. Angrisani *et al.* [43] consider the processing and queuing delays of open-source routers and propose a measurement method that can distinguish the time interval between packets that remain in the input and output queues and characterize the effective routing process of the packets.

B. RESEARCH ON REQUEST RESPONSE DELAY

Request response delay refers to the time from the instant that a request is sent to the time that a response is received. Caching is a method that is often used to improve users' QoE. Caching uses on-path routers to cache its forwarded packets and return them to users when packets are requested again, which avoids lengthy routes and reduces the request response time.

The most primitive caching methods, such as the cache and replacement algorithm of [47], consider less overhead. Under this caching algorithm, each router needs to maintain an information table and other nodes. Some researches suggests that only local neighbor information needs to be maintained but only in a hierarchical network topology. In fact, it is effective to maintain the cache information of the global network at data center, and when data are requested, query it directly from data center, so that the communication overhead is smaller, and can implement global data integration [48].

To improve the speed of data lookup and service return, the storage mechanism of cached content is often considered. The cache mechanism based on content popularity is effective. By storing the content packets with high request frequency at routers closest to users, the nearest acquisition of data and the minimum request response time can be achieved. This mechanism has been adopted in [49] and is especially suitable for multimedia networks with large data volumes.

Luo *et al.* [50] consider energy consumption and service delay of IDC in the context of a sharp rise in demand for cloud computing services. In this paper, a novel two-stage design and eco-IDC algorithm are proposed to dynamically schedule the workload using the time diversity of the electricity price and execute it on the IDC server via input queues. The evaluation results show that this method can significantly reduce the energy of IDC and guarantee the request response delay.

Shen *et al.* [51] use a geographically distributed cloud to minimize the service latency and incorporate privacy protection. In this system, the resource allocation scheme enables the distributed cloud server to cooperatively allocate servers to users under load balancing, which minimizes the request response delay.

C. RESEARCH ON LOAD BALANCING

A data center is the data processing center of the entire network. Similar to the sink node in a wireless sensor network, the data center has a very heavy workload. By improving the network architecture, data can be localized as much as possible, which can effectively reduce the workload of the data center. For example, extend the network architecture from cloud computing to a multilayered edge network. Mobile edge computing are excellent computing paradigms that enable users to access services anytime and anywhere [52]. The main idea is to transfer the data calculation from the network center to the edge and use the IoT devices at the edge to store services, which reduces the delay and overhead caused by long-distance requests to the network center.

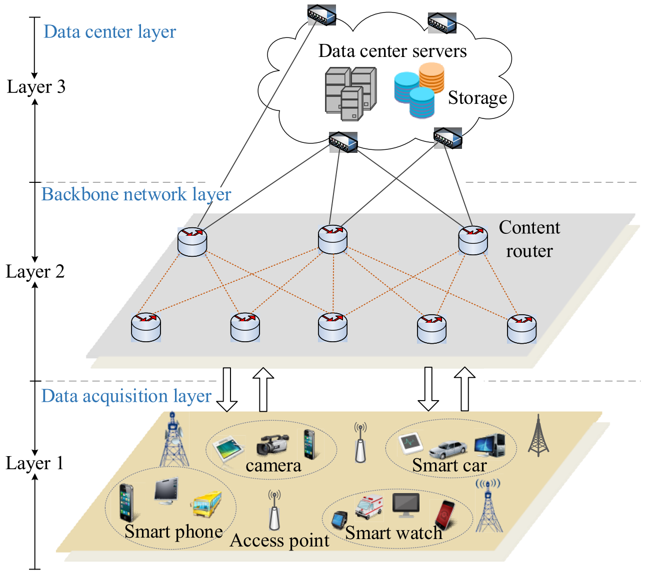

FIGURE 1. Network architecture.

III. SYSTEM MODEL AND PROBLEM STATEMENT

A. NETWORK ARCHITECTURE

The network architecture of this paper is illustrated in Figure 1, and the network roles involved in this system include data center servers, content routers, base stations, gateways and underlying data gathering devices. According to the division of various equipment and the data interaction between devices, the entire network can be divided into three layers, namely, data acquisition layer, backbone network layer and data center layer.

The data acquisition layer constructs a network basic information system, which is composed of numerous IoT devices that are distributed in real-life scenarios, including various industrial and civilian monitoring and sensing devices, such as smart phones, smart cars, laptops, and monitoring probes [5], [9], [52]. These data gathering devices are referred to Big Data Collectors (BDCs) in the cloud network and entities in the Cyber Physical System (CPS). When these BDCs obtain data from the surrounding environment, they report data to the Data Center (DC) via multi hop routes. Previously, these data sensing devices primarily consisted of sensor nodes deployed at specific locations. With the rapid development of the IoT, an increasing number of mobile terminals and smart devices are employed for data gathering.

The backbone network layer is the transport intermediary between IoT devices and DC, which consists of content routers with certain storage and computing power [14], [28], [52]. These content routers not only complete data routing but also perform temporary data caching and simple data calculations when necessary. The data processing rate of these devices substantially affects the delay and throughput of the entire network.

The data center layer is located at the top of the network. All packets are eventually aggregated into the data center, analyzed and processed by the data center servers, and placed in storage. Devices in this layer are large devices with vast storage capacity and powerful computing capabilities deployed by the Service Provider (SP) [39]. These devices orchestrate data as services and provide to users in response to service requests. In the background of big data, the speed, capacity and type of network data are exponentially increasing, while data center hosts all data processing and service routes in the network; thus, its workload is very heavy, which causes network performance indicators, such as delay, jitter, packet loss rate, reliability, and throughput, to deteriorate [39], [40].

The standardized description of the network can be defined as a directed graph $\mathbb{G} = \{R, L\}$, where R is the set of all routers, and L is an ordered set of two-tuple groups of elements in R. Assume N routers in the network, $R = \{r_1, r_2, \ldots, r_N\}$, $|R| = N$, L represents the communication links between pairs of routers. For the link between two routers, $L = (L_S, L_E)$, where L_S is the starting point of the path and L_E is the ending point of the path. In addition, the set of data packets is $\mathbb{D}$, $\mathbb{D} = \{D_1, D_2, \ldots, D_K\}$, and $|\mathbb{D}| = K$. The data are aggregated and cached locally in the form of service packets, and the set of service packets is $\mathbb{S} = \{S_1, S_2, \ldots, S_M\}$, $|\mathbb{S}| = M$.

B. SERVICE AGGREGATION MODEL

Service aggregation refers to the process of filtering erroneous, redundant and invalid information from the raw data, aggregating the valid information of multiple packets, and encapsulating this information into services that can be directly applied by users [39]. Therefore, service aggregation is similar to data fusion, that is, the higher is the correlation between data, the higher is the aggregation rate. The classic lossless step-by-step multi hop aggregation model is adopted in this paper [38], [40]. Assuming that the set of data packets to be aggregated of router r_i is $\mathbb{C}_{r_i}$, $\left|\mathbb{C}_{r_i}\right| = K$, the *j-th* data packet in $\mathbb{C}_{r_i}$ is $D_{r_i}^{j}$, and the aggregation rate is $\mathcal{E}$, then the aggregation result of the K packets is

$$\mathcal{A}\left(\mathbb{C}_{r_i}^{Agg}\right) = \mathcal{E}\sum_{j=1}^{K}\left(D_{r_i}^{j}\right) \tag{1}$$

C. PROBLEM STATEMENT

1) MINIMIZE REQUEST RESPONSE DELAY

Request Response Delay (RRD) is a direct reflection of the computing and processing abilities of the network and an important indicator that affects users' Quality of Experience (QoE). RRD is associated with the working efficiency of data center servers, the total amount of data stored in the network, and the location of storage. In general, the higher is the processing rate of servers, the smaller is the total amount of storage data, and the closer is the storage location to users, the smaller is the RRD. Dividing the network life cycle into K time slots, the number of requests in a time slot is M (M is different for different slots), and the response delay of the *j-th* request in the *i-th* time slot is d_i^j. Minimize the request response delay as

$$Min(\mathfrak{D}_{RRD}) = Min\left(\sum_{i=1}^{K}\sum_{j=1}^{M} d_i^j\right) \tag{2}$$

2) REDUCE THE DATA AMOUNT TO BE TRANSMITTED

For routers, the amount of data transmitted is the sum of the data that it forwards. For the data center, the data amount is the sum of the data that it receives. In general, the larger is the amount of data that is transmitted in the network, the more extensive is the data content, the richer are the data types, and the greater is the transmission consumption. Therefore, in big data networks, the key is to improve the data quality, that is, reduce the proportion of redundant data. Assuming that the set of routers is R, $R = \{r_1, r_2, \ldots, r_N\}$, $|R| = N$, the total data amount transmitted by r_i is Π_{tot}^{i}, and the effective data amount is Φ_i. The reduced transmission data amount is

$$Min(\mathfrak{I}_D) = Min\left[\sum_{i=1}^{N}\left(\left|\frac{\Pi_{tot}^{i} - \Phi_i}{\Pi_{tot}^{i}}\right|\right)\right] \tag{3}$$

3) BALANCE NETWORK LOAD

Network load refers to the traffic inherited by network relays and the number of users hosted by the data center. Thus, balancing the network load refers to avoiding the extreme situation of overloading a device by distributing tasks to many devices. Assuming that there are N routers in the network, the data amount assumed by r_i is $\mathcal{R}_i$ and the average data amount of routers is $\mathcal{R}_{avg}$, then balance network load can be expressed as:

$$Min\left(\mathfrak{Q}_E\right) = Min\left(\sum_{i=1}^{N}\left|\mathcal{R}_i - \mathcal{R}_{avg}\right|\right) \tag{4}$$

Therefore, the research objectives are expressed as follows:

$$\begin{cases} Min(\mathfrak{D}_{RRD}), & \mathfrak{D}_{RRD} = \displaystyle\sum_{i=1}^{K}\sum_{j=1}^{M} d_i^j \\ Min\left(\mathfrak{I}_D\right), & \mathfrak{I}_D = \displaystyle\sum_{i=1}^{N}\left(\left|\frac{\Pi_{tot}^{i} - \Phi_i}{\Pi_{tot}^{i}}\right|\right) \\ Min\left(\mathfrak{Q}_E\right), & \mathfrak{Q}_E = \displaystyle\sum_{i=1}^{N}\left|\mathcal{R}_i - \mathcal{R}_{avg}\right| \end{cases} \tag{5}$$

IV. DESIGN OF SAQD

In this section, a detailed description of the SAQD scheme is provided. First, we introduce the terms involved in SAQD. Second, the idea of the scheme is described. Last, we present specific implementation algorithms. The terms used in SAQD are as described as follows:

The forwarding queue refers to the queue used by routers to temporarily store packets that need to be forwarded. If a router receives a new data packet during the process of sending data packets, the new packet is placed in the forwarding queue first.

Queuing delay refers to the time that a packet remains in the forwarding queue. The time starts from the time when the packet enters the queue and ends when the packet is forwarded from the queue.

The differentiating threshold is a mark that is used in the forwarding queue to distinguish between forwarding packets and aggregating packets. In the forwarding queue, packets before the differentiating threshold are forwarding packets, and packets after the threshold are aggregating packets.

The ending mark, which is applied in the forwarding queue to mark the last aggregated packet in the queue, is utilized to distinguish between the aggregated packets and the packets to be aggregated.

The main idea of the scheme is to perform the service aggregation using the queuing delays of packets while ensuring a normal enqueue and dequeue order of the forwarding queue. When the packet is waiting to be sent out in the queue, we merge it with other packets in the queue, remove their invalid data, and encapsulate the valid data into a service stored in the router's local storage. In this case, the queuing delays of packets are reasonably employed for data processing but the amount of data transmitted is reduced.

The complete routing and computing of packets in the backbone network layer are investigated in SAQD, including relay selection between two-hop routes, determination of the differentiating threshold, and specific service aggregation considerations.

A. RELAY SELECTION AND PACKETS ASSIGNMENT

In previous schemes, routers typically select the node that is closest to them as the next hop, which minimizes the propagation delay of a single hop; however, the queue length of the selected relay node is unknown. Especially in data-intensive networks, numerous packets may be waiting to be processed; thus, the total latency may not be minimal. Unlike previous studies, in this paper we determine the next hop based on the forwarding queue length of the relay node.

In this paper, we select the node with the smallest queue length as the next hop and consider the distance as the second basis. Assuming that router r_i needs to send data, r_i selects the next hop from the set Q_F. Q_F is a relay candidate set formed by nodes within the communication scope of r_i; the nodes in the set are sorted according to the queue length. Nodes with a smaller queue length are arranged in the preceding traversal position in the set. When the queue length is equal, the node closer to r_i is arranged in front. Q_F is constantly updated, and when a node is assigned a packet, its queue length is increased by 1. Conversely, if a node sends a packet, its queue length is reduced by 1. Selecting a relay node based on the minimum queue length can minimize the queuing delay. However, for the case in which the queue length of the selected node may be small but the node is located far from the current node, the propagation delay may increase at this point. For this case, we set the distance threshold ϕ_d. If the distance between the selected node and the current node exceeds ϕ_d, a large propagation delay may be generated, then a new node is reselected from Q_F. The ϕ_d of different routers may not be the same, which is related to the launch radius and channel quality. When selecting a relay node, we choose the node with the smallest queue length within the ϕ_d range of the current node.

Assume that the sending node is R_{sd}, the distance threshold is ϕ_d, the candidate set of relay nodes is Q_F, and the total number of nodes in the set is $|Q_F|$. If the maximum length of the forwarding queue is $\mathcal{L}_o$, the current queue length is ℓ, the distance to R_{sd} is γ_i^{sd}, and the selected relay node is R_{re}, then the selection algorithm of R_{re} can be illustrated by Algorithm 1.

Algorithm 1 Selection of Relay Nodes

```
For each $R_{sd}$ ready to send data Do
  Initialize $i = 1$, $R_{re} = $ null, $\mathrm{Ix\_rel} = 0$
  While $i \leq |Q_F|$ Do
    If $\gamma_i^{sd} \leq \phi_d$ and $\mathcal{L}_o - \ell_i \geq 1$ Do
      Let $R_i$ to be the relay node $R_{re}$
      $\mathrm{Ix\_rel} = i$
      Dequeue packet and send to $R_{re}$
      $\ell_i = \ell_i + 1$, $\ell_{sd} = \ell_{sd} - 1$
      $j = \mathrm{Ix\_rel} + 1$;
      While $j \leq |Q_F|$ Do
        If $\ell_j < \ell_{\mathrm{Ix\_rel}}$ or ($\ell_j = \ell_{\mathrm{Ix\_rel}}$ and $\gamma_{\mathrm{j}}^{sd} \leq \gamma_{\mathrm{Ix\_rel}}^{sd}$) Do
          Let $R_{\mathrm{j}}$ in the j-1 th position of $Q_F$
          j++;
        End if
        Else
          break
        End else
      End while
      Let $R_{\mathrm{Ix\_rel}}$ in the j-1 th position of $Q_F$
      Quit
    End if
    Else
      i++;
    End else
  End while
End for
```

For the set Q_F, which consists of candidate nodes that have been sorted by queue length, R_{sd} traverses the first node in the set when selecting the relay node, that is, the node with the smallest queue length. As long as the distance of the node is within ϕ_d, it is selected as the relay node. After the relay node is selected, the queue length of the sending node and the receiving node is updated, and the relay node is prioritized and relocated in the correct position in the set according to the updated queue length. Therefore, the complexity of the relay node selection algorithm is very small. Since Q_F is

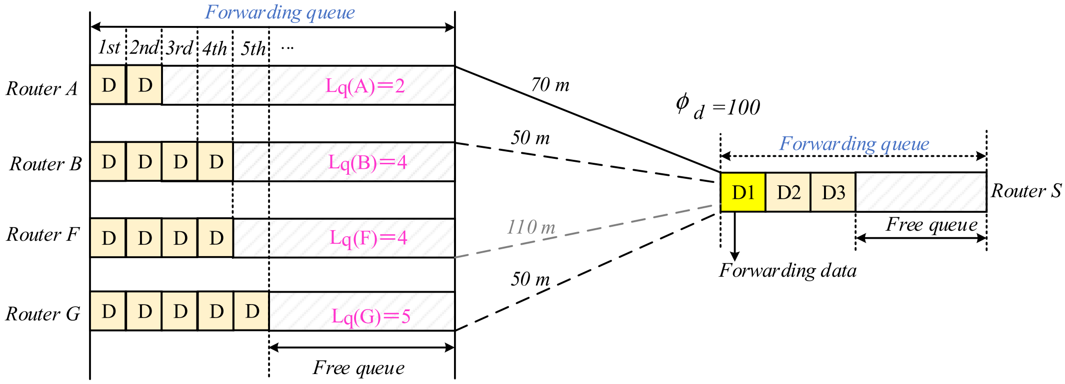

FIGURE 2. Relay selection between two-hop routes.

a well-ordered set, as long as the distance satisfies the requirements, generally the first node (the worst case may traverse to 2 or 3) is the relay node, and the complexity at this time is constant and can be disregarded. The main overhead of Algorithm 1 is the position movement of nodes whose queue length is less than the selected relay node. Assuming that M nodes exist before the relay node, the overhead of this part is O (M). M is very small because the relay node is selected from nodes in front of the set and its queue length is only increased by 1 after updating the queue length. Thus, the relay node generally only moves a few locations based on the existing position, that is, only a small number of nodes behind it need to move. Therefore, the time and space complexity of Algorithm 1 is small.

As shown in Figure 2, assuming that the current node is Router S, the three data packets D1, D2, and D3 exist in the forwarding queue of Router S, where D1 is the packet to be sent, and Router S requires the distance of the relay node to be limited to 100 m. Four nodes exist in the communication range of Router S, namely, Router A, Router B, Router F and Router G, and the length of their forwarding queues are 2, 4, 4, and 5. These four nodes construct the relay candidate set Q_F of Router S and are prioritized by the queue length from small to large. When Router S wants to send D1, it traverses Q_F first and then finds Router A. Since Router A is within the allowed distance from Router S (the distance between them is 70 m), Router A is selected as the receiving node of D1. After D1 is sent from the forwarding queue of Router S, the queue length of Router S is decremented by 1, while the queue length of Router A is increased by 1. When Router S sends the second packet D2, Router B is traversed first. Although Router F and Router B have the same queue length, Router F is farther than Router B. Therefore, Router F is placed after Router B. Similarly, after confirming that the distance of Router B is within the allowable range, D2 is sent to Router B. Router S sends D3, at which point Router F is traversed first. Because its distance from the current node exceeds the maximum distance of 100 m, so the data transfer to Router F may cause a larger propagation delay. We discard Router F and reselect a new node from Q_F. This relay selection method makes packets that have a smaller queuing delay. Since the node receives one packet each time, its queue length is increased, which prevents individual or part nodes from frequently receiving packets. This packets assignment is fairly balanced.

B. DETERMINATION OF DIFFERENTIATING THRESHOLD

Routers use the First Come First Serve (FCFS) queue scheduling method. Service aggregation is performed at each hop by utilizing the dequeue time of packets. Before the aggregation, the differentiated threshold is continuously adjusted and determined according to the number of packets in the queue. Based on the differentiated threshold, packets in the queue are classified, forwarding packets are located before the threshold, and aggregating packets are placed after the differentiating threshold. The ending mark is the location of the last aggregating packet.

Assume that the number of packets in a service aggregation is N_{tot}, the data transmission rate of routers is $\Re^i_{tr}$, the data aggregation rate is $\Re_{\text{ag}}$ ($\Re^i_{tr} < \Re_{\text{ag}}$), the data amount of each aggregating packet is Π^i_{raw}, the data amount of each forwarding packet is Π^i_{fow} ($\Pi^i_{fow} < \Pi^i_{raw}$), and the differentiating threshold is established to satisfy the following equation:

$$\sum_{i=1}^{\mathcal{F}_\Theta}\left(\frac{\Pi^i_{fow}}{\Re^i_{tr}}\right) - \frac{\sum_{j=1}^{N_{tot}-\mathcal{F}_\Theta}\Pi^j_{raw}}{\Re_{ag}} > \vartheta \tag{6}$$

where ϑ is an allowable difference between the forwarding time and the aggregation time. Since the differentiating threshold is the boundary of the forwarding packets and aggregating packets, the total data transmission time of all forwarding packets in one aggregation must be greater than the service aggregation time of aggregating packets. Thus, aggregating packets can be forwarded from the queue in accordance with normal order.

Assume that the current queue length is N_{tot}, the data transmission rate of routers is $\Re_{tr}$, the data aggregation rate is $\Re_{ag}$, the data amount of each forwarding packet is Π^i_{fow}, the data amount of each aggregating packet is Π^j_{raw}, the ending mark is E_Ξ, and the differentiating threshold algorithm can be described by Algorithm 2.

Algorithm 2 Determination of Differentiating Threshold

```
Let $\mathfrak{F}_\Theta = N_{tot}/2$
Loop: Initialize $\Pi_{tot} = 0$, $T_{tot}^{fd} = 0$, $T_{tot}^{agg} = 0$
Initialize variables $i = 1, j = \mathfrak{F}_\Theta + 1$
While $i \le \mathfrak{F}_\Theta$ Do
    $T_{tot}^{fd} = T_{tot}^{fd} + \frac{\Pi_{fow}^{i}}{\Re_{tr}^{i}}$
End while
While $j \le N_{tot}$ Do
    $\Pi_{tot} = \Pi_{tot} + \Pi_{raw}^{j}$
End while
$T_{tot}^{agg} = \frac{\Pi_{tot}}{\Re_{ag}}$
While $T_{tot}^{fd} - T_{tot}^{agg} \le \vartheta$ Do
    If $(N_{tot} - \mathfrak{F}_\Theta \ge 2)$ Do
        $\mathfrak{F}_\Theta = \mathfrak{F}_\Theta + 1$
    End if
    Else
        $N_{tot} = N_{tot} - 1$
        $\mathfrak{F}_\Theta = N_{tot}/2$
    End else
    If ($\mathfrak{F}_\Theta \ge E_\Xi$ and $N_{tot} \ge 3$) Do
        Goto Loop
    End if
    Else
        $\mathfrak{F}_\Theta = 0$
        Quit
    End else
End while
Output: the differentiating threshold $\mathfrak{F}_\Theta$
```

The determination process is described as follows: according to the current queue length, half of the queue length is set to the initial value of the differentiating threshold. If the queue length is odd, the differentiating threshold is set to the smaller integer. Based on the differentiating threshold, packets before it are classified as forwarding packets, and the total forwarding time required to transmit these packets is computed. Packets after the differentiating threshold are classified as aggregating packets, the total amount of raw data of these packets are calculated, and the total aggregation time required to aggregate these packets is computed according to the data aggregation rate of routers. If the data forwarding time is sufficient for support service aggregation with the initial differentiating threshold, then the initial value is the differentiating threshold of this aggregation. If the data forwarding time is less than the required service aggregation time, the differentiating threshold is adjusted based on the initial value. When adjusting, increase the value of the differentiation threshold, reduce the number of aggregating packets, increase the number of forwarding packets, and calculate the new data forwarding time and service aggregation time after adjustment until the time requirement is satisfied. Continuously reduce the number of aggregating packets. When the number of aggregating packets is less than 2, service aggregation cannot be completed. In this case, we adjust the total number of packets, that is, some packets in the current aggregation are reserved for the next aggregation. Repeat this process after adjusting the total number of packets until the value of the differentiation threshold is determined. When the network data are small (e.g., the total number of packets is less than 3 due to at least one forwarding packet and two aggregating packets), or the differentiating threshold is adjusted to be less than the ending mark, then the service aggregation condition is not satisfied. Therefore, we end the algorithm and set the value of the differentiating threshold to 0. In Algorithm 2, if the value of the differentiating threshold is 0, no aggregation occurs. Otherwise, the corresponding threshold value of the aggregation is output.

In Algorithm 2, the intermediate value of the queue length is established as the initial value of the differentiating threshold. Assuming that the current queue length is N, in the best case, the value of the differentiating threshold is determined as the initial value, and the data lookup overhead is $\mathrm{O}(1)$. The forwarding time and aggregation time for this initial value should be calculated. Thus, each packet in the queue needs to be traversed, the overhead for this part is $\mathrm{O}(\mathrm{N})$, and the complexity in the best case is $\mathrm{O}(1) * \mathrm{O}(\mathrm{N}) = \mathrm{O}(\mathrm{N})$. In the general case, the value of the differentiating threshold is continuously increased to reduce the aggregating packets. The differentiating threshold is gradually increased from the initial value to the N-1th packet, and the data lookup overhead at this time is $\mathrm{O}\left(\frac{\mathrm{N}}{2} - 1\right)$. Similarly, the forwarding time and aggregating time are calculated for each value, the cost of this part is $\mathrm{O}(\mathrm{N})$, and the complexity in the general case is $\mathrm{O}\left(\frac{\mathrm{N}}{2} - 1\right) * \mathrm{O}(\mathrm{N}) = \mathrm{O}\left(\mathrm{N}^2\right)$. In the worst case, the total number of packets is adjusted. Meanwhile, the process of setting the initial value is repeated, and in the least ideal case, it is necessary to adjust the number of aggregating packets again based on the initial value. When the number of total packets is adjusted to three, the overhead is maximized, coupled with the overhead of calculating the forwarding time and aggregation time, the complexity in the worst case is $\mathrm{O}(\mathrm{N} - 3) * O\left(\frac{\mathrm{N}}{2} - 1\right) * \mathrm{O}(\mathrm{N}) = \mathrm{O}\left(\mathrm{N}^3\right)$. Although the complexity of Algorithm 2 is cubic, its actual cost and complexity are not high because the value of N is not large. Routers in a data sparse network have fewer packets, the value of N is small. In a data-intensive network, the queue length of routers is limited, and the value of N has an upper bound. What' more, the worst case in Algorithm 2 is an extreme case; thus, we can consider that the time complexity of Algorithm 2 is quartered.

C. ON-PATH SERVICE AGGREGATION

Service aggregation begins after determining the differentiating threshold. Service aggregation is a recently proposed new technology that relies on proposed data cleansing and data fusion technologies [38], [45], [46]. As shown in Figure 3,

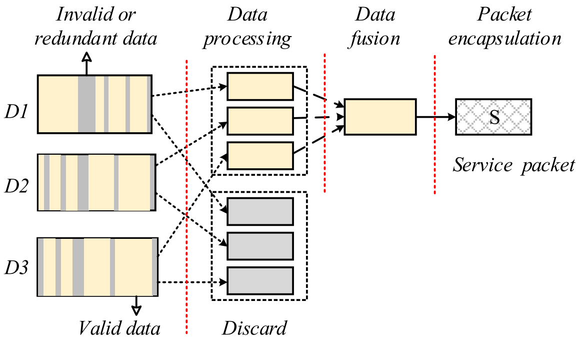

FIGURE 3. Service aggregation of multiple packets.

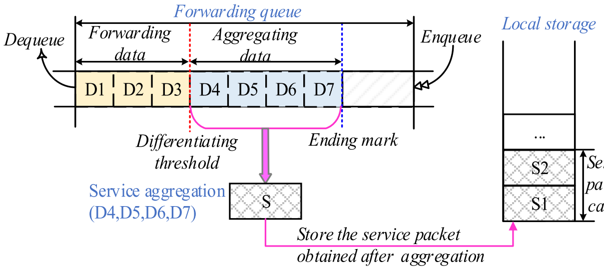

FIGURE 4. Service aggregation using queuing delay.

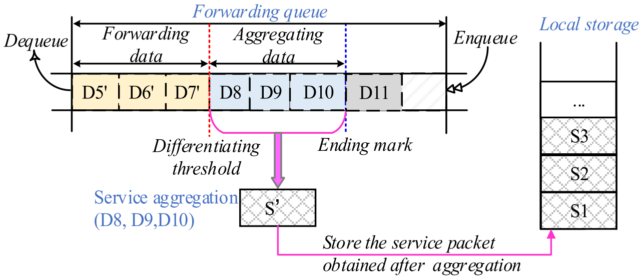

FIGURE 5. Part service aggregation using queuing delay.

the first step in service aggregation is similar to data filtering: remove redundant data from multiple packets and invalid data in a single packet, and then retain the valid data in original packets for subsequent data forwarding and data fusion. As a result, the number of packets that are transferred in the network does not decrease after the service is aggregated but the data amount per packet and the total amount of data that is transferred are reduced. The second step in service aggregation is to correlate the valid data retained by multiple packets and fuse the data, which is similar to data fusion. The final step in service aggregation is to encapsulate the merged data in the form of a service and store the service packet in the local storage.

When cached locally, the service packet is divided into multiple content chunks, chunks of the same service are stored in consecutive units, and chunks of different services are stored in different units [48], [49]. However, compared with the data center, the storage capacity of routers is limited. When the storage space is full, the Least Recently Unused (LRU) replacement algorithm is employed. The stored services have a storage record table in the data center and the corresponding router. The difference is that the data center has a global service storage mapping table, which is used to record the mapping between the service and the corresponding storage router. The storage record table at each router is used to record the mapping between services and chunks. We focus on the study of queuing delay utilization in this paper, and for specific implementations of caching, please refer to one of our other papers [40].

The previous idea can be briefly described by Figure 4. As shown in Figure 4, the router has a forwarding queue and local storage, where the forwarding queue is used to store data to be forwarded, and the local storage is used to store service packets obtained by service aggregation. Both the queue length and the storage capacity are limited. The forwarding queue of the router has a total of 7 packets. The data transmission time of the first three forwarding packets is sufficient for the aggregation of the subsequent four aggregating packets according to Algorithm 2. Thus, the value of the differentiating threshold is set to 3, and the ending mark is marked after D7. The first three forwarding packets are sent according to the FCFS rules, and subsequent D4, D5, D6, and D7 perform service aggregation, with the remaining empty queue for data reception. After the four data packets are aggregated, service packet S is obtained, service S is stored in the local storage of the router, and the record is added to the storage mapping table of the router and the data center. In the router's storage, S1, S2,. . . are stored. If the new service packet S is too large, or the storage space is full, the service with sufficient storage space and the most recently unused service will be replaced by S.

After D1-D3 are sent and D4-D7 are aggregated, processed D4-D7 are obtained and named D4′-D7′. D4′ is transmitted to the next hop, and D5′-D7′ locate in front of the forwarding queue. As shown in Figure 5, the data transmission time of D5′, D6′, and D7′ are only sufficient for aggregating D8, D9, and D10, and D11 is left for the next aggregation. In the SAQD scheme, the queuing delay and processing packets are different each time, which is determined according to the actual queue length and the queuing situation. As shown in Figure 5, D8, D9, and D10 are aggregated to obtain S′, and S′ is stored locally.

Since the data correlation of different packets differs, the degree of aggregation and the time required for aggregation may vary. In a network with a suitable channel quality and high transmission bandwidth, the data transmission time will be shorter. As a result, in some cases, forwarding data packets has been transferred but service aggregation remains in process. As shown in Figure 6, the last packet of forwarding data has dequeued but D11 continues to aggregate with D12 and D13. In response to this situation, we introduce the allowed waiting time λ, that is, we tolerate a λ waiting time if the forwarding data are sent but the service aggregation is not completed. If the service aggregation can be completed within λ, then transmit D11 after the aggregation. If the service aggregation is underway after the λ time, then the

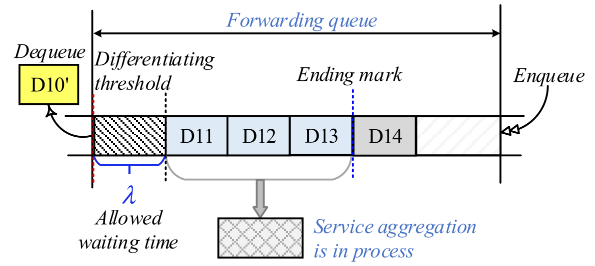

FIGURE 6. **Allowed waiting time for unfinished aggregation.**

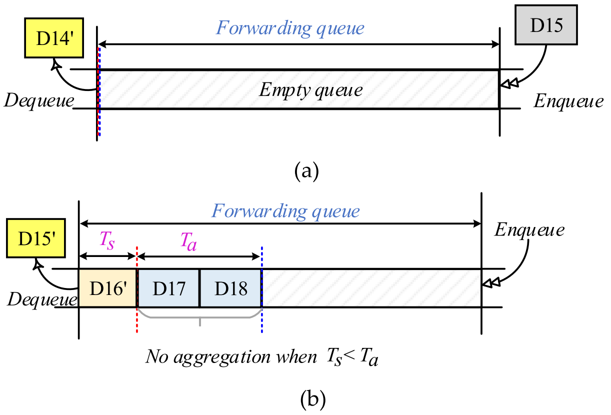

FIGURE 7. **Cases with no service aggregation. (a) Empty forwarding queue. (b) Queuing data are insufficient to support aggregation.**

aggregation is canceled, and D11 is sent with the form of a raw data packet.

In two cases, the raw packets are directly sent without service aggregation, as shown in Figure 7. When the queue is empty, the new enqueue packets are directly sent without waiting for subsequent packets. In Figure 7(a), after D14′ dequeues, the queue is empty, at which point the D15 is directly sent to the next route. At this time, no service aggregation occurs. As shown in Figure 7 (b), D16′, D17, and D18 exist in the queue, and the transmission time of D16′ is not sufficient for supporting aggregation of D17 and D18. From another perspective, if D16′ and D17 are employed as forwarding packets, D18 cannot be aggregated because it is a single packet. When the queue packets are few and cannot be divided into forwarding packets and aggregation packets, we directly send original packets as in other solutions.

Assume that the current queue length is N_{tot}, the ending mark is E_Ξ, the differentiating threshold is $\mathcal{F}_\Theta$, the allowable waiting time is λ, the current waiting time is Υ_w, the set of forwarding packets is $\mathbb{C}_{\text{for}}$, and the set of aggregating packets is $\mathbb{C}_{\text{agg}}$; then the service aggregation algorithm can be described by Algorithm 3. When the queue length attains a certain value ($N^i_{tot} - E_\Xi \geq \varrho$), the service aggregation is started. The differentiating threshold is determined according to Algorithm 2, forwarding packets and aggregating packets are divided by the differentiating threshold, and forwarding packets are sent according to Algorithm 1; alternatively, aggregating packets are sent after the service aggregation and their data amount is reduced, which are based on Algorithm 1. When the aggregation fails or no aggregation occurs, the original packets are sent. The complexity of Algorithm 3 depends on Algorithm 1 and Algorithm 2.

Algorithm 3 Service Aggregation With Queuing Delay

```
For each $R_i$ in network Do
  Initialize $\mathbb{C}_{\text{for}}$ = null, $\mathbb{C}_{\text{agg}}$ = null, $\Upsilon_w = 0$
  While $N^i_{tot} - E_\Xi \geq \varrho$ Do
    Compute $\mathcal{F}_\Theta$ use Algorithm 2
    If $\mathcal{F}_\Theta != 0$ Do
      Add packets before $\mathcal{F}_\Theta$ to $\mathbb{C}_{\text{for}}$
      Add packets after $\mathcal{F}_\Theta$ to $\mathbb{C}_{\text{agg}}$
        Loop1: For each packet in $C_{\text{for}}$ Do
          Select the receive node use Algorithm 1
          Dequeue packet and remove it from $\mathbb{C}_{\text{for}}$
        End for
        For packets in $\mathbb{C}_{\text{agg}}$ Do
          Perform service aggregation
          Let $\mathbb{C}_{\text{agg}}$ = null
        End for
        If $\mathbb{C}_{\text{for}}$ =null and $\mathbb{C}_{\text{agg}}$! =null Do
          Wait time $\lambda$ for service aggregation
        End if
        If $\mathbb{C}_{\text{agg}}$! =null and $\Upsilon_w \geq \lambda$ Do
          Stop service aggregation
          Move original data packets in $\mathbb{C}_{\text{agg}}$ to $\mathbb{C}_{\text{for}}$
          Goto Loop1
        End if
        Else
          Add processing data packets in $\mathbb{C}_{\text{agg}}$ to $\mathbb{C}_{\text{for}}$
          Goto Loop1
          Update ending mark $E_\Xi$
        End else
      End if
      Else
        Dequeue all packets use Algorithm 1
     End else
  End while
End for
```

V. PERFORMANCE ANALYSIS AND EXPERIMENT RESULTS

A. EXPERIMENT SETUP

In this section, the effectiveness of SAQD is analyzed from the aspects of data volume, delay, power consumption and response time. The FSR scheme is selected as the experimental comparison scheme. The FSR scheme is a classic network communication scheme. In this FSR scheme, routers act as transmission relays to transfer original packets to the data center, and packets are not processed or cached during the transmission. In queue scheduling, routers in the FSR scheme use the FCFS principle while enqueuing and dequeuing packets.

TABLE 1. Experimental parameters.

Parameter	Value
Network span	1000 m*1000 m
Number of routers	56
Number of data center	1
Deployment of routers	Randomly
Packet size	10-20 KB
Data frequency	20000/50000/100000 packets/s
Data transmission rate	10 Mbps
Routing strategy	Shortest queue/Shortest routing
Queue length of routers	10
Communication radius	100 m
Storage capacity	10.45 MB
Storage replacement	Least Recently Unused

The experimental scenario is a 1000 m*1000 m planar network, with 1 data center and 56 content routers that are randomly deployed in the network, and the launch radius of routers is 100 m. Similar to [36] and [40], the network parameters are set as follows: (1) Data size of each packet ranges from 10 KB-20 KB. When the data volume is less than 10 KB, the data packet is merged with other data packets. When the data volume is larger than 20 KB, the data packet is split into multiple small packets. (2) The network has three different data generation frequencies. In general, the network produces 20,000 packets per second. For comparison, we also consider the data frequency of 50,000 packets per second and 100,000 packets per second. The network medium has a data transfer rate of 10 Mbps. (3) SAQD and FSR adopt different routing strategies, SAQD adopts the shortest queue length priority strategy, and FSR adopts the shortest distance first algorithm. (4) The maximum queue length of routers is 10, and the storage capacity is 10.45 MB. When the storage space reaches the upper limit, the Least Recently Unused (LRU) replacement algorithm is employed. The specific parameter values are given in Table 1.

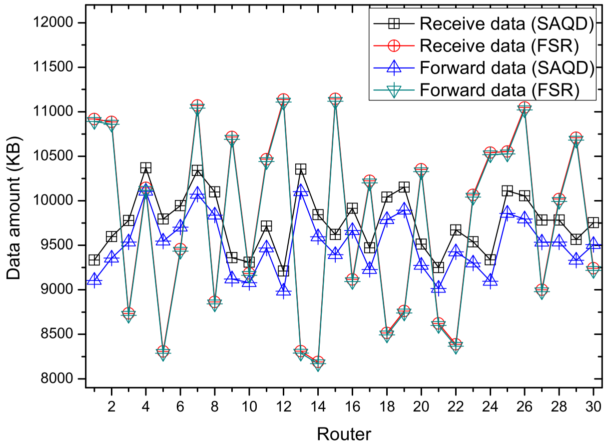

FIGURE 8. Data amount received and forwarded by routers.

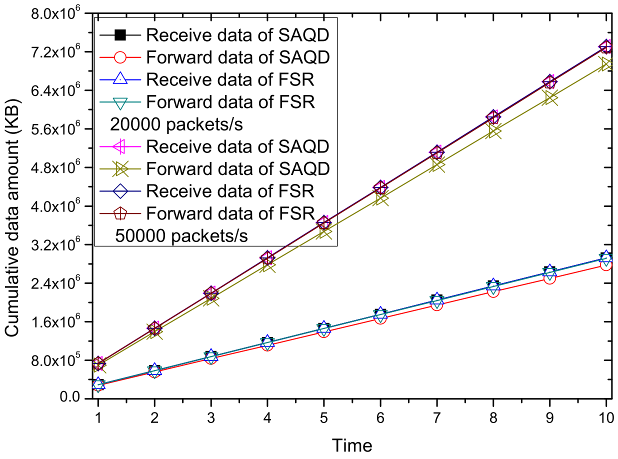

FIGURE 9. Cumulative data amount with an increase in time.

B. DATA AMOUNT

Figures 8-11 show the data load of routers for different data frequencies. When 20,000 packets are generated in the network per second and the data volume of each packet ranges from 10 KB-20 KB, the amount of data carried by part routers with SAQD and FSR is depicted in Figure 8. A comparison of the two schemes reveals that the data volume of routers in the SAQD scheme is more balanced and maintained between 9000 KB and 10500 KB. In the FSR scheme, the maximum data volume of routers is approximately 11000 KB, the minimum data volume of routers is approximately 8000 KB, and the data volume gap between routers is significantly larger than that of SAQD. This finding is attributed to the data allocation mechanism of the two schemes. In the SAQD scheme, routers preferentially select nodes with smaller queue length as relays, and nodes with smaller queue lengths receive less data. In addition, the queue lengths of nodes are constantly updated. Therefore, nodes in this routing strategy will not receive data frequently, which avoids an unbalanced amount of data. The FSR scheme selects the relay node based on the shortest distance. Because the deployment of routing nodes in the network is often static, the selected relay node for the same router in different times tends to be fixed in this routing algorithm, which causes nodes that deploy in more dense areas to carry more data forwarding volume. A comparison of the different types of data within the same scheme in Figure 8 indicates that forwarding data are significantly less than the received data in the SAQD scheme. SAQD enables routers to locally aggregate data for services, which eliminates some invalid information in packets and reduces the amount of data. In the FSR scheme, routers forward original data packets to the next hop without data processing; thus, the amount of data remains unchanged.

Select three different data frequencies—20,000 packets, 50,000 packets and 100,000 packets per second—and randomly choose 11 routers distributed in different regions; the number of received packets of these 11 routers with SAQD and FSR is shown in Table 2. The distribution of packets is consistent with the amount of data in Figure 8. In the

TABLE 2. **Receiving packets of routers in SAQD and FSR.**

	20000 packets/s		50000 packets/s		100000 packets/s	
	SAQD	FSR	SAQD	FSR	SAQD	FSR
R1	642	627	1634	1532	3370	3899
R2	668	725	1677	1803	3434	3966
R3	629	759	1775	1896	3238	3881
R4	698	574	1666	1934	3355	2889
R5	668	559	1763	1513	3295	2994
R6	653	765	1658	1795	3385	3073
R7	678	624	1654	1484	3418	3030
R8	648	702	1609	1809	3309	2954
R9	689	586	1708	1500	3392	3839
R10	689	600	1620	1864	3325	3923
R11	646	700	1645	1505	3290	3854

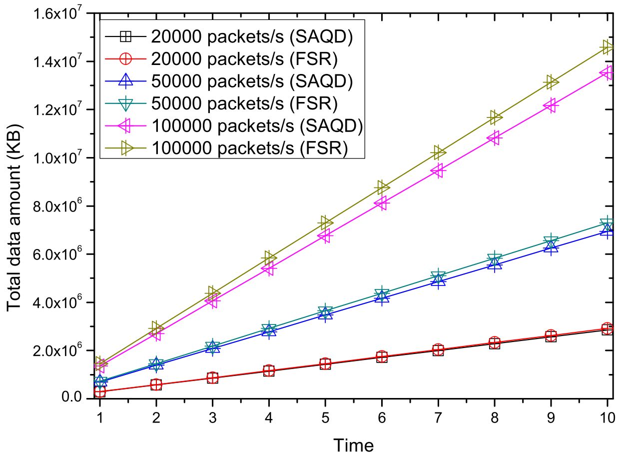

FIGURE 10. **Total data amount for different data frequencies.**

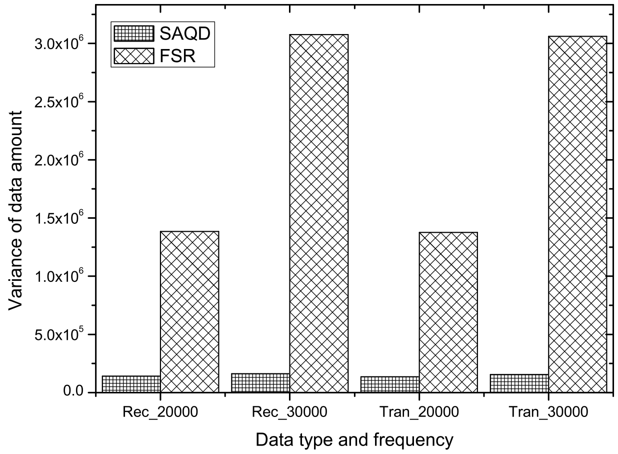

FIGURE 11. **Variance of data amount for SAQD and FSR.**

SAQD scheme, the number of packets of each router is similar, and the data load of routers is relatively balanced. As the data frequency increases, an increasing difference exists in the packets of routers with the FSR scheme, and the number of packets of the most heavily loaded router is 34.96% more than the number of packets of the lighter router.

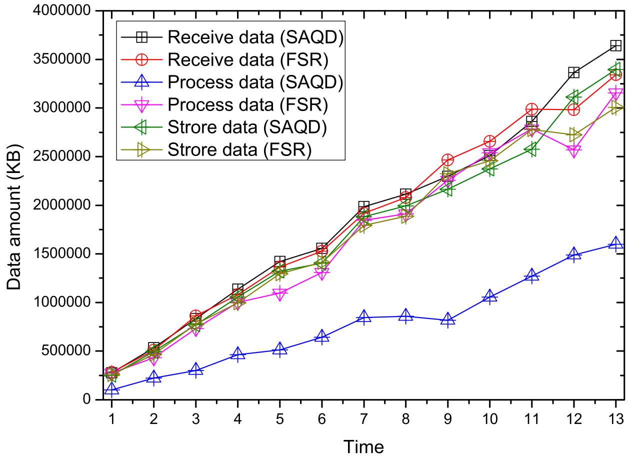

FIGURE 12. **Data amount of data center for SAQD and FSR.**

Figure 9 shows the amount of received data and forwarded data of routers with an increase in time. The amount of data linearly increases with time, and the higher is the data frequency, the faster is the growth. A comparison of the SAQD and FSR schemes reveals that the difference in the receiving data is minimal. In the SAQD scheme, however, the transmission data redundancy is reduced after service aggregation. Thus, the amount of forwarding data is less than the amount of receiving data, which indicates that the on-path service aggregation can effectively reduce the amount of data that is forwarded.

Figure 10 shows a growth comparison of the total data amount for the SAQD and FSR schemes. Using three different data frequencies, the total data volume for SAQD is less than 7.28% of the total data volume for FSR, especially with an increase in data frequency; the total data of the two schemes is significantly different.

Figure 11 shows the variance of the data amount of routers in the SAQD and FSR schemes. Compared with the FSR scheme, the variance of the data amount in the SAQD scheme is very small. When the data frequency is 20,000 packets/s, the variance of the received data of SAQD is only 10.32% of the FSR, and the variance of the transmitted data is 9.87% of the FSR. With an increase in data frequency, the variance difference is more distinct. When the data frequency is 30,000 packets/s, the variance of the received data of SAQD is 5.3% of FSR, and the variance of the transmitted data is 5.05% of FSR.

Figures 12-13 describe the data load of the data center. Figure 12 shows the amount of data that is received, processed and stored by the data center. The amount of received data and stored data in the SAQD and FSR schemes are similar. However, the amount of data processed by the data center for the SAQD scheme is substantially less than that of the FSR scheme. In the SAQD scheme, data processing and partial requests can be directly processed by routers without being sent to the data center, which reduces the workload of the data center. As intuitively demonstrated in Figure 13,

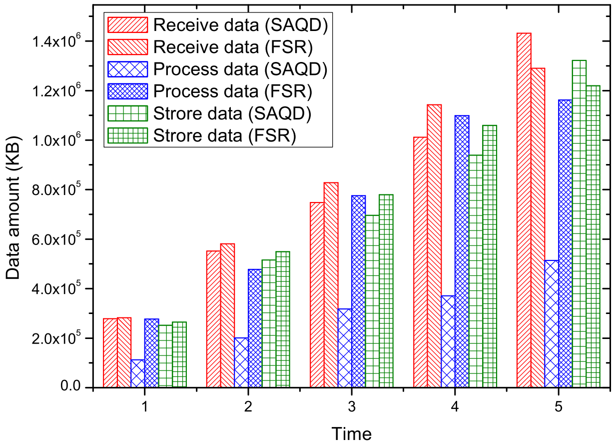

FIGURE 13. **Data amount of data center for different data types.**

compared with the FSR scheme, the amount of data processed by the data center in the SAQD scheme is reduced from 55.8%-66.26%.

Based on Figures 8-13, compared with the conventional FSR scheme, the SAQD scheme proposed in this paper is effective in balancing the data volume of routers and reducing the workload of the data center.

C. DELAY

Assume that the SAQD and FSR schemes are applied in the same network environment, that is, the propagation speed and the transmission bandwidth are equivalent, the data transmission rate is 2.5 KB/ms, and the data aggregation rate is 3.5 KB/ms. The delay of the two schemes is shown in Figures 14-18.

Figure 14 shows the propagation delay for the two schemes. Since the propagation delay of each hop is related to the transmission distance, as shown in Figure 14, the single hop propagation delay in the FSR scheme is significantly lower than that in SAQD because the FSR scheme employs the shortest distance routing algorithm. Although the single-hop propagation delay of the FSR is smaller than that of SAQD, the number of routing hops of FSR is substantially greater than that of SAQD. As shown in the experimental example in Figure 14, the packet is transmitted to the data center at the 21st hop in SAQD, while that packet is transferred to the data center after 34 hops in FSR. For the total propagation delay, the difference between SAQD and FSR is minimal.

The transmission delay of SAQD and FSR is illustrated in Figure 15. Unlike the propagation delay, the transmission delay is affected by the transmission bandwidth and the amount of data. When the transmission bandwidths are equivalent, the transmission delay is primarily related to the data amount. With the SAQD scheme, data are aggregated once per route; thus, the amount of data decreases as the number of aggregations increases. Therefore, the one-hop transmission delay in the SAQD scheme is smaller than that in the FSR scheme. In addition, the transmission delay is closely related to the number of route hops. Since the FSR scheme has a larger number of relay hops, the total transmission delay is much higher than that in SAQD.

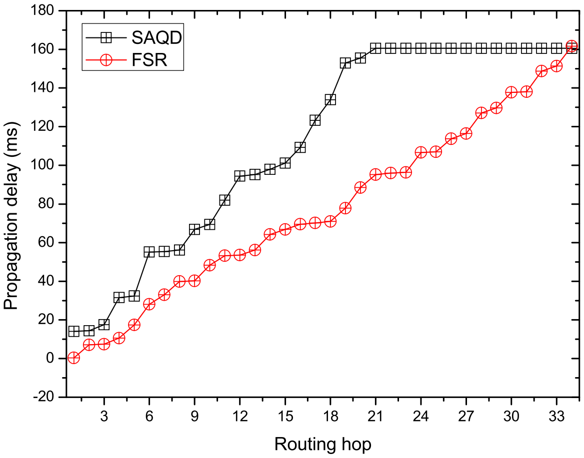

FIGURE 14. **Propagation delay for different routing hops.**

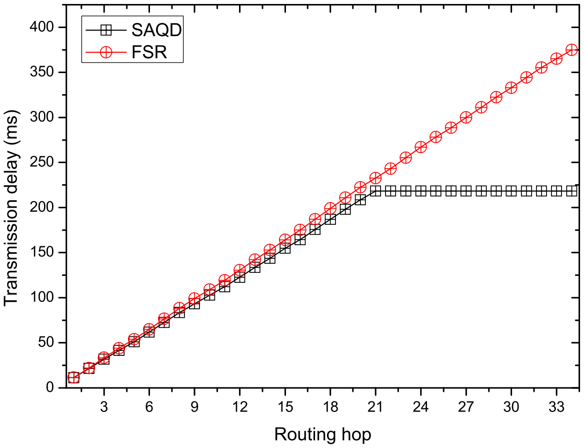

FIGURE 15. **Transmission delay for different routing hops.**

Figure 16 depicts the queuing delay. In the SAQD scheme, we apply the shortest queue-first algorithm to select the relay node; thus, the queuing delay of nodes is small. The shortest distance is applied in the FSR scheme but the current node does not know the queue length of the next hop. Therefore, the queue delay of each hop in the FSR scheme is uncertain. For the queuing delay, whether it is single-hop delay or end-to-end delay, the SAQD scheme is better than the FSR scheme.

The delay includes propagation delay, transmission delay and queuing delay. Figure 17 shows the total delay at different distances from the data center. First, the data transmission delay increases with the distance from the data center, with smaller latency closer to data center and larger latency farther from the data center. Second, compared with the FSR scheme, the total delay of the SAQD scheme in this paper decreases as the transmission distance increases. SAQD has substantial advantages in transmission delay and queuing delay.

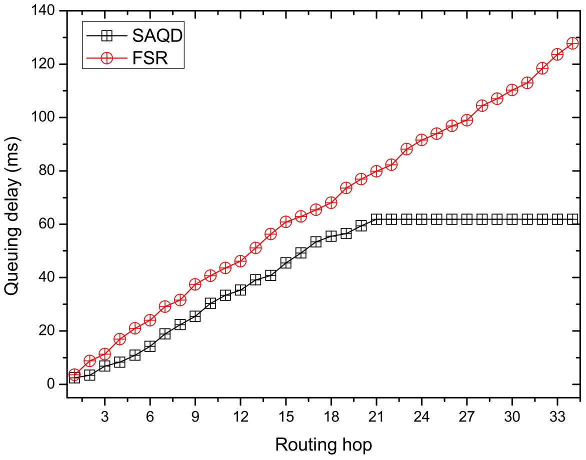

FIGURE 16. Queuing delay for different routing hops.

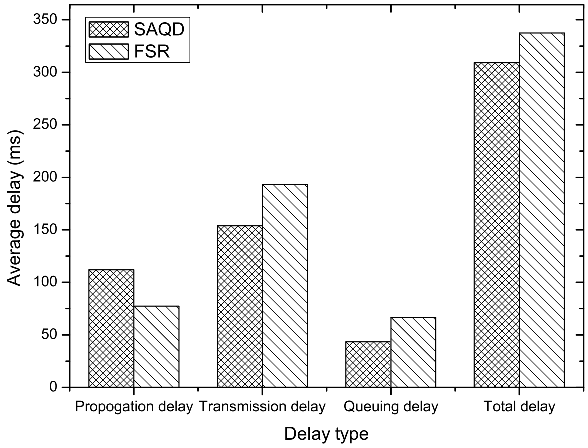

FIGURE 18. Average delay of SAQD and FSR.

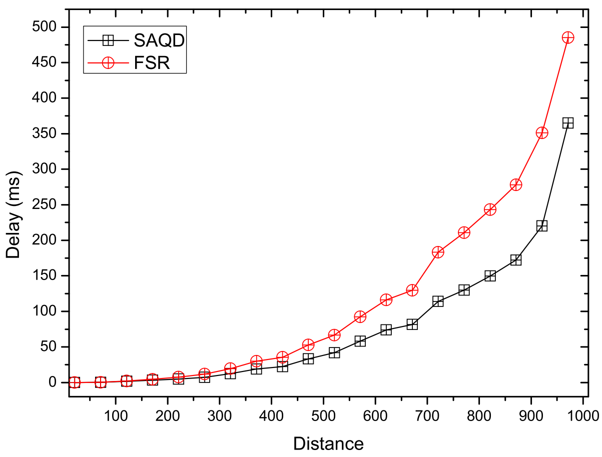

FIGURE 17. Delay at different distances from the data center.

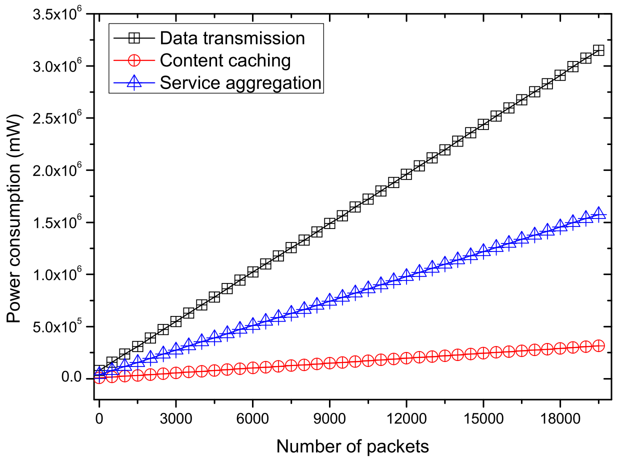

FIGURE 19. Power consumption of different data operations with SAQD.

The average delay of SAQD and FSR is compared in Figure 18. The results show that the average propagation delay of SAQD is 30.92% higher than that of FSR, whereas the average transmission delay of SAQD is 25.81% lower than that of FSR, and the queuing delay is 53.4% lower than that of FSR. The delay of SAQD is 9.15% lower than that of FSR.

D. POWER CONSUMPTION

In this experimental scenario, we set the data transfer consumption of routers to 1 mW/KB, the data storage consumption to 0.1 mW/KB, the service aggregation consumption to 0.5 mW/KB, the data storage consumption of the data center to 0.1 mW/KB, and the data processing consumption of the data center to 0.5 mW/KB. Simultaneously, 20,000 packets are produced per second, and the time elapsed from the generation of these 20,000 packets to receipt by the data center is a time slot.

In the SAQD scheme, the consumption of routers includes the consumption of data transmission, the consumption of data storage and the consumption of service aggregation. The power consumption of these three parts is shown in Figure 19, in which the consumption of data transmission is the most substantial, the consumption of service aggregation is ranked second, and the power consumption of data storage is relatively small. The consumption of routers increases in proportion to the number of data packets.

Figure 20 shows the total consumption of routers in the SAQD and FSR schemes. In the FSR scheme, routers only have data transmission consumption. Since the data are processed by each hop in the SAQD scheme, the amount of data transmitted is continuously reduced. As a result, the transmission consumption in the SAQD scheme is considerably lower than that of the FSR scheme. Considering the overhead of data storage and data aggregation, the total power consumption in the SAQD scheme is higher than that in the FSR.

Although Figure 20 shows that the total consumption in SAQD is higher than that in FSR, as shown in Figure 21, the power consumption of routers in SAQD is more balanced, and the gap between maximum consumption and minimum consumption is small. In the FSR scheme, the consumption of routers varies substantially, and the maximum power consumption is several times the minimum.

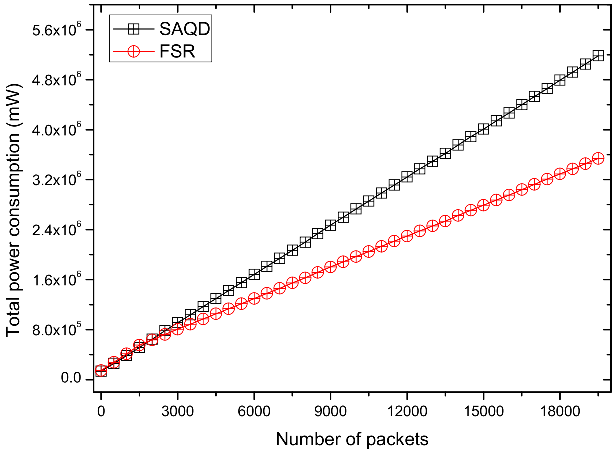

FIGURE 20. Total power consumption of SAQD and FSR.

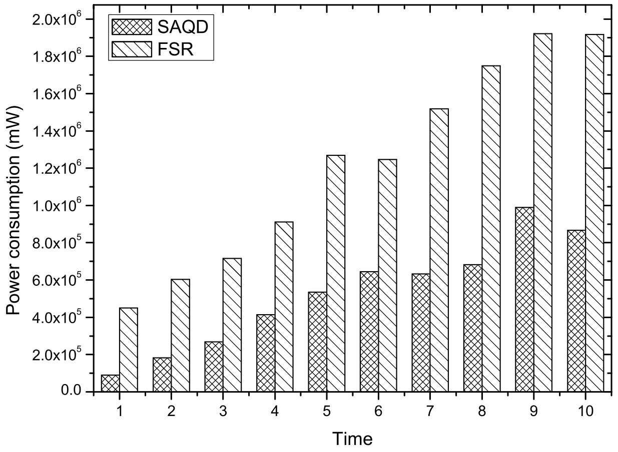

FIGURE 23. Power consumption of the data center with an increase of time.

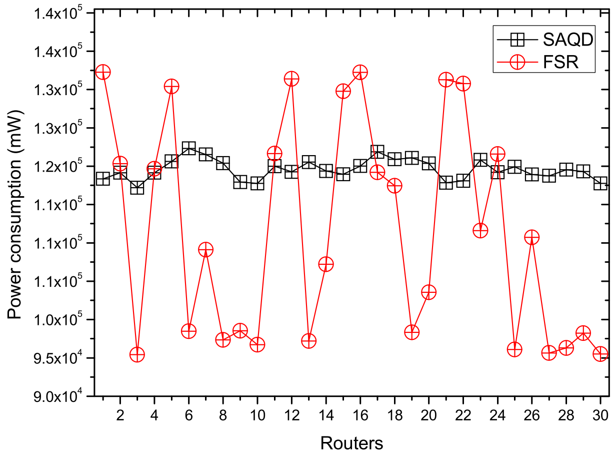

FIGURE 21. Power consumption of routers.

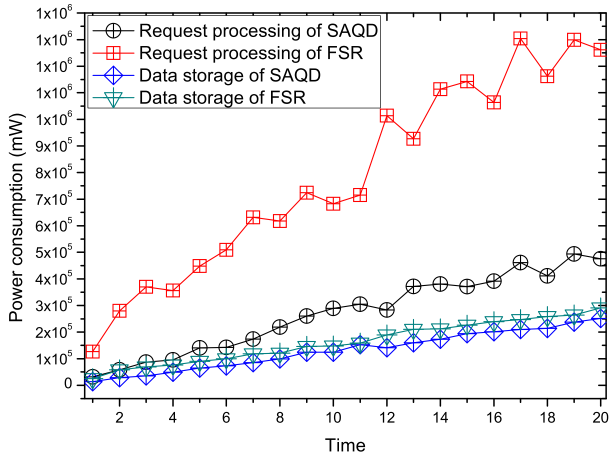

FIGURE 22. Power consumption of data center.

Figure 22 and Figure 23 illustrate the power consumption of the data center for the SAQD and FSR schemes. The power consumption of the data center includes data storage overhead and request response consumption. Figure 22 shows the data storage and requests processing overhead of the two schemes. The difference in data storage overhead between SAQD and FSR is minimal, and SAQD is slightly smaller than FSR. In terms of request processing, the overhead of SAQD is considerably smaller than that of FSR because some requests in SAQD can be satisfied by routers locally, and all requests in the FSR scheme must be processed by the data center.

Figure 23 shows the total power consumption of the data center. The power consumption of the data center with SAQD is 48.52%-69.72% lower than that of the FSR scheme. This finding indicates that SAQD can reduce the load of the data center by the local aggregation and storage of routers.

E. REQUEST RESPONSE DELAY

In this experimental scenario, 10,000 requests are sent by users per second. The requested data volume ranges from 10-20 KB, 40-60 KB, and 80-100 KB. Routers return data if the requested data are cached in their local storage in the SAQD scheme; otherwise, the data are returned from the data center. In the FSR scheme, all requests are handled by the data center.

Figure 24 illustrates the request response time for different requested data volumes. A comparison of the two schemes, for the same requested data volume, the response delay for the SAQD scheme is lower than that of the FSR scheme. As the amount of requested data increases, the delay difference between the two schemes is larger. In the SAQD scheme, as the proportion of data returned by routers increases, the response delay gradually decreases, which indicates that returning data from routers has a lower delay than the returning data from the data center.

As the amount of requested data increases, the response latency also increases, as shown in Figure 25. In the FSR scheme, each request needs to be sent to the data center via a lengthy route, and the data are returned to users again via the same lengthy route. Therefore, the larger is the amount of data that is requested, the higher is the delay. In the SAQD scheme, routers disperse the data processing pressure of the data center, and the delay growth becomes moderate.

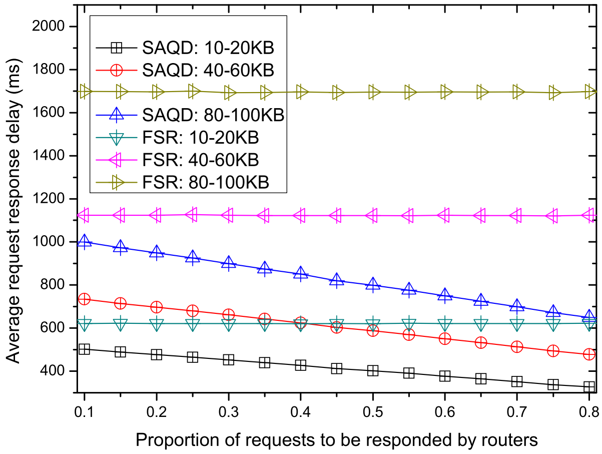

FIGURE 24. Average request response delay for different proportions of requests to be responded by routers.

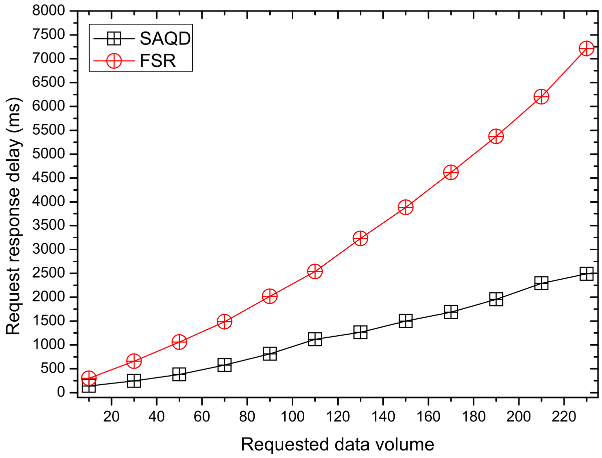

FIGURE 25. Request response delay for different requested data volumes.

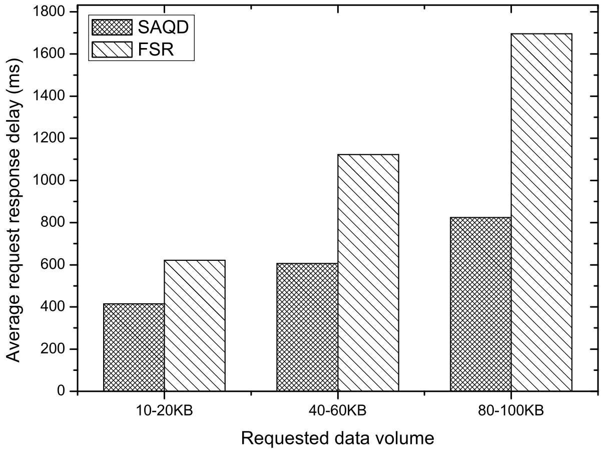

FIGURE 26. Average request response delay of SAQD and FSR.

The average delay in the SAQD scheme is approximately 34.55%∼47.28% of the FSR scheme.

Figure 26 shows the average request response delay. Compared with the FSR scheme, when the requested data volume ranges from 10-20 KB, the SAQD scheme reduces the request response delay by 33.33%. When the requested data volume ranges from 40-60 KB, the SAQD scheme reduces the request delay by 46.08%. When the requested data volume ranges from 80-100 KB, the SAQD scheme reduces the request delay by 51.41%.

VI. CONCLUSIONS

In this paper, we propose a queuing delay utilization scheme for on-path service aggregation (SAQD), which can be applied to service-oriented computing networks to reduce the data amount and improve users' Quality of Experience. This study is the first attempt by researchers to exploit the queuing delay of packets to improve network performance. A service aggregation algorithm is provided in the scheme, which distinguishes forwarding packets and aggregating packets according to the real-time queue length, processes aggregating packets by utilizing the transmitting time of forwarding packets, and caches the services obtained after service aggregation in the local storage of routers. A queue-length-based relay selection and data assignment algorithm is proposed to ensure the load balancing of routers. The experimental results demonstrate that SAQD has distinct advantages in service-oriented networks with an immense volume of data traffic, especially in reducing request response delay and load balancing.

REFERENCES

[1] T. Qiu, R. Qiao, M. Han, A. K. Sangaiah, and I. Lee, "A lifetime-enhanced data collecting scheme for the Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 132–137, Nov. 2017.

[2] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation offloading for service workflow in mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 12, pp. 3317–3329, Dec. 2015.

[3] X. Liu, Q. Yang, J. Luo, B. Ding, and S. Zhang, "An energy-aware offloading framework for edge-augmented mobile RFID systems," *IEEE Internet Things J.*, to be published. doi: 10.1109/JIOT.2018.2881295.

[4] Y. Ren, W. Liu, T. Wang, X. Li, N. N. Xiong, and A. Liu, "A collaboration platform for effective task and data reporter selection in crowdsourcing network," *IEEE Access*, to be published. doi: 10.1109/ACCESS.2019.2897062.

[5] H. Teng, Y. Liu, A. Liu, N. N. Xiong, Z. Cai, T. Wang, X. Liu, "A novel code data dissemination scheme for Internet of Things through mobile vehicle of smart cities," *Future Gener. Comput. Syst.*, vol. 94, pp. 351–367, May 2019.

[6] A. Liu, J. Min, K. Ota, and M. Zhao, "Reliable differentiated services optimization for network coding cooperative communication system," *Int. J. Comput. Syst. Sci. Eng.*, vol. 33, no. 4, pp. 235–250, 2018.

[7] H. Teng, W. Liu, T. Wang, A. Liu, X. Liu, and S. Zhang, "A cost-efficient greedy code dissemination scheme through vehicle to sensing devices (V2SD) communication in smart city," *IEEE Access*, vol. 7, pp. 16675–16694, 2019. doi: 10.1109/ACCESS.2019.2895899.

[8] X. Deng, J. Luo, L. He, Q. Liu, X. Li, and L. Cai, "Cooperative channel allocation and scheduling in multi-interface wireless mesh networks," *Peer-to-Peer Netw. Appl.*, vol. 12, no. 1, pp. 1–12, 2019.

[9] X. Sun and N. Ansari, "EdgeIoT: Mobile edge computing for the Internet of Things," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 22–29, Dec. 2016.

[10] S. Zhang, G. Wang, M. Z. A. Bhuiyan, and Q. Liu, "A dual privacy preserving scheme in continuous location-based services," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4191–4200, Oct. 2018.

[11] Y. Liu, A. Liu, X. Liu, and X. Huang, "A statistical approach to participant selection in location-based social networks for offline event marketing," *Inf. Sci.*, vol. 480, pp. 90–108, Apr. 2019.

[12] J. Tan *et al.*, "An adaptive collection scheme based matrix completion for data gathering in energy-harvesting wireless sensor network," *IEEE Access*, vol. 7, no. 1, pp. 6703–6723, 2019.

[13] M. Huang, A. Liu, N. N. Xiong, T. Wang, and A. V. Vasilakos, "A low-latency communication scheme for mobile wireless sensor control systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 2, pp. 317–332, Feb. 2019.

[14] Y. Liu, M. Ma, X. Liu, N. Xiong, A. Liu, and Y. Zhu, "Design and analysis of probing route to defense sink-hole attacks for Internet of Things security," *IEEE Trans. Netw. Sci. Eng.*, to be published. doi: 10.1109/TNSE.2018.2881152.

[15] A. Liu and S. Zhao, "High-performance target tracking scheme with low prediction precision requirement in WSNs," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 29, no. 4, pp. 270–289, 2018.

[16] W. Qi *et al.*, "Minimizing delay and transmission times with long lifetime in code dissemination scheme for high loss ratio and low duty cycle wireless sensor networks," *Sensors*, vol. 18, no. 10, p. 3516, 2018.

[17] X. Xiang, W. Liu, N. N. Xiong, H. Song, A. Liu, and T. Wang, "Duty cycle adaptive adjustment based device to device (D2D) communication scheme for WSNs," *IEEE Access*, vol. 6, no. 1, pp. 76339–76373, 2018.

[18] X. Liu, Y. Liu, N. Zhang, W. Wu, and A. Liu, "Optimizing trajectory of unmanned aerial vehicles for efficient data acquisition: A matrix completion approach," *IEEE Internet Things J.*, to be published. doi: 10.1109/JIOT.2019.2894257.

[19] Y. Ren, Y. Liu, N. Zhang, A. Liu, N. N. Xiong, and Z. Cai, "Minimum-cost mobile crowdsourcing with QoS guarantee using matrix completion technique," *Pervasive Mobile Comput.*, vol. 49, pp. 23–44, Sep. 2018.

[20] X. Ju *et al.*, "An energy conserving and transmission radius adaptive scheme to optimize performance of energy harvesting sensor networks.," *Sensors*, vol. 18, no. 9, p. E2885, 2018.

[21] T. Li, S. Tian, A. Liu, H. Liu, and T. Pei, "DDSV: Optimizing delay and delivery ratio for multimedia big data collection in mobile sensing vehicles," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3474–3486, Oct. 2018.

[22] T. Wang, G. Zhang, M. Z. A. Bhuiyan, A. Liu, W. Jia, and M. Xie, "A novel trust mechanism based on fog computing in Sensor–Cloud System," *Future Gener. Comput. Syst.*, to be published. doi: 10.1016/j.future.2018.05.049.

[23] X. Liu *et al.*, "Construction of large-scale low-cost delivery infrastructure using vehicular networks," *IEEE Access*, vol. 6, no. 1, pp. 21482–21497, 2018.

[24] B. Huang, W. Liu, T. Wang, X. Li, H. Song, and A. Liu, "Deployment optimization of data centers in vehicular networks," *IEEE Access*, to be published. doi: 10.1109/ACCESS.2019.2897615.

[25] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, "A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing," *IEEE Internet Things J.*, to be published. doi: 10.1109/JIOT.2018.2870288.

[26] T. Li, K. Ota, T. Wang, X. Li, Z. Cai, A. Liu, "Optimizing the coverage via the UAVs with lower costs for information-centric Internet of Things," *IEEE Access*, vol. 7, pp. 15292–15309, 2019. doi: 10.1109/ACCESS.2019.2894172.

[27] Y. Liu, A. Liu, N. Zhang, X. Liu, M. Ma, and Y. Hu, "DDC: Dynamic duty cycle for improving delay and energy efficiency in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 131, pp. 16–27, Apr. 2019.

[28] X. Luo, Y. Lv, M. Zhou, W. Wang, and W. Zhao, "A laguerre neural network-based ADP learning scheme with its application to tracking control in the Internet of Things," *Pers. Ubiquitous Comput.*, vol. 20, no. 3, pp. 361–372, 2016.

[29] J. Ren, Y. Zhang, N. Zhang, D. Zhang, and X. Shen, "Dynamic channel access to improve energy efficiency in cognitive radio sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 5, pp. 3143–3156, May 2016.

[30] Buuuk. (2012). *WeatherLah iPhone Application*. [Online]. Available: http://itunes.apple.com/us/app/weatherlah/id411646329?mt=8

[31] Cisco. (2018). *Internet of Things Market Forecast*. [Online]. Available: http://postscapes.com/internet-of-things-market-size

[32] Cisco. (2018). *Internet of Things Market Forecast*. [Online]. Available: http://postscapes.com/internet-of-things-market-size

[33] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Inf. Syst.*, vol. 47, pp. 98–115, Jan. 2015.

[34] W. Zhao, "Performance optimization for state machine replication based on application semantics: A review," *J. Syst. Softw.*, vol. 112, pp. 96–109, Feb. 2016.

[35] W. Zhao, P. M. Melliar-Smith, and L. E. Moser, "Fault tolerance middleware for cloud computing," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2010, pp. 67–74.

[36] J. Xu, X. Liu, M. Ma, A. Liu, T. Wang, and C. Huang, "Intelligent aggregation based on content routing scheme for cloud computing," *Symmetry*, vol. 9, no. 1, p. 221, 2017.

[37] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, "Mobile cloud computing: A survey, state of art and future directions," *Mobile Netw. Appl.*, vol. 19, no. 2, pp. 133–143, 2014.

[38] J. Abdelaziz, M. Adda, and H. Mcheick, "Toward service aggregation for edge computing," *Procedia Comput. Sci.*, vol. 94, pp. 424–428, Aug. 2016.

[39] Z. Ding *et al.*, "Orchestrating data as a services-based computing and communication model for information-centric Internet of Things," *IEEE Access*, vol. 6, pp. 38900–38920, 2018.

[40] M. Huang *et al.*, "A services routing based caching scheme for cloud assisted CRNs," *IEEE Access*, vol. 6, pp. 15787–15805, 2018.

[41] J. Gui and K. Zhou, "Cellular throughput optimization by game-based power adjustment and outband D2D communication," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, p. 254, Oct. 2018.

[42] J. Tan *et al.*, "A low redundancy data collection scheme to maximize lifetime using matrix completion technique," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 9, p. 5, 2019. doi: 10.1186/s13638-018-1313-0.

[43] L. Angrisani, G. Ventre, L. Peluso, and A. Tedesco, "Measurement of processing and queuing delays introduced by an open-source router in a single-hop network," *IEEE Trans. Instrum. Meas.*, vol. 55, no. 4, pp. 1065–1076, Aug. 2006.

[44] G. Z. Papadopoulos, V. Kotsiou, A. Gallais, and P. Chatzimisios, "Low-power neighbor discovery for mobility-aware wireless sensor networks," *Ad Hoc Netw.*, vol. 48, pp. 66–79, Sep. 2016.

[45] L. A. Villas, A. Boukerche, H. S. Ramos, H. A. F. de Oliveira, R. B. de Araujo, and A. A. F. Loureiro, "DRINA: A lightweight and reliable routing approach for in-network aggregation in wireless sensor networks," *IEEE Trans. Comput.*, vol. 62, no. 4, pp. 676–689, Apr. 2013.

[46] X. Li, A. Liu, M. Xie, N. Xiong, Z. Zeng, and Z. Cai, "Adaptive aggregation routing to reduce delay for multi-layer wireless sensor networks," *Sensors*, vol. 18, no. 4, p. 1216, 2018.

[47] L. Dong, D. Zhang, Y. Zhang, and D. Raychaudhuri, "Optimal caching with content broadcast in cache-and-forward networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–5.

[48] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3665–3677, Oct. 2014.

[49] M. F. Majeed, M. N. Dailey, R. Khan, and A. Tunpan, "Pre-caching: A proactive scheme for caching video traffic in named data mesh networks," *J. Netw. Comput. Appl.*, vol. 87, pp. 116–130, Jun. 2017.

[50] J. Luo, L. Rao, and X. Liu, "Temporal load balancing with service delay guarantees for data center energy cost optimization," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 775–784, Mar. 2014.

[51] Q. Shen, X. Liang, X. Shen, X. Lin, and H. Y. Luo, "Exploiting geo-distributed clouds for a E-health monitoring system with minimum service delay and privacy preservation," *IEEE J. Biomed. Health Informat.*, vol. 18, no. 2, pp. 430–439, Mar. 2014.

[52] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of Internet of Things," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Jan./Mar. 2018.

MINGFENG HUANG is currently pursuing the master's degree with the School of Computer Science and Engineering, Central South University, China. Her research interests include services-based networks, crowd sensing networks, and wireless sensor networks.

WEI LIU received the Ph.D. degree in computer application technology from Central South University, China, in 2014. He is currently an Associate Professor and a Senior Engineer with the School of Informatics, Hunan University of Chinese Medicine, China. He has published more than 20 papers in related fields. His research interests include software engineering, data mining, and medical informatics.

TIAN WANG received the B.Sc. and M.Sc. degrees in computer science from Central South University, in 2004 and 2007, respectively, and the Ph.D. degree from the City University of Hong Kong, in 2011. He is currently an Associate Professor with Huaqiao University, China. His research interests include wireless sensor networks, social networks, and mobile computing.

HOUBING SONG (M'12–SM'14) received the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, USA, in 2012. In 2017, he joined the Department of Electrical, Computer, Software, and Systems Engineering, Embry-Riddle Aeronautical University, Daytona Beach, FL, USA, where he is currently an Assistant Professor and the Director of the Security and Optimization for Networked Globe Laboratory (SONG Lab, www.SONGLab.us). He served on the faculty of West Virginia University, from 2012 to 2017. In 2007, he was an Engineering Research Associate with the Texas A&M Transportation Institute. He is the Editor of four books: *Smart Cities: Foundations, Principles, and Applications*, (Hoboken, NJ: Wiley, 2017); *Security and Privacy in Cyber-Physical Systems: Foundations, Principles, and Applications*, (Chichester, U.K.: Wiley-IEEE Press, 2017); *Cyber-Physical Systems: Foundations, Principles and Applications*, (Boston, MA: Academic Press, 2016); and *Industrial Internet of Things: Cybermanufacturing Systems*, (Cham, Switzerland: Springer, 2016). He is the author of more than 100 articles. His research interests include cyber-physical systems, cybersecurity and privacy, the Internet of Things, edge computing, big data analytics, unmanned aircraft systems, connected vehicle, smart and connected health, and wireless communications and networking.

Dr. Song is a Senior Member of ACM. In 2016, he was a First Recipient of the Golden Bear Scholar Award, which is the highest campus-wide recognition for research excellence with the West Virginia University Institute of Technology. He serves as an Associate Technical Editor for the *IEEE Communications Magazine*.

XIONG LI received the master's degree in mathematics and cryptography from Shanxi Normal University, China, in 2009, and the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications, China, in 2012. He is currently an Associate Professor with the School of Computer Science and Engineering, Hunan University of Science and Technology, China. He has published more than 70 referred journal papers in his research interests, which include cryptography, information security, and cloud computing security. He has served as a TPC Member of several international conferences on information security and a Reviewer for more than 30 ISI indexed journals. He was a recipient of the 2015 *Journal of Network and Computer Applications* Best Research Paper Award. He is currently an Editor of *Telecommunication Systems* and *KSII Transactions on Internet and Information Systems*.

ANFENG LIU received the M.Sc. degree in computer science and the Ph.D. degree in computer science from Central South University, China, in 2002 and 2005, respectively, where he is currently a Professor with the School of Computer Science and Engineering. His major research interests include cloud computing, fog computing, and wireless sensor networks. He is a member (E200012141M) of the China Computer Federation.

• • •