# An F5G-oriented Fast link Recovery Mechanism for High-Concurrency Services


Zhang Mengxi
*State Grid Economic and Technological Research Institute Co.,LTD.*
Beijing China
zhang_mxi@163.com

Jin Guangxiang
*State Grid Economic and Technological Research Institute Co.,LTD.*
Beijing China
jinguangxiang@chinasperi.sgcc.com.cn

Cui Zijian
*Beijing University of Posts and Telecommunications*
Beijing China
zijiancui@bupt.edu.cn

Li Aonan
*Beijing University of Posts and Telecommunications*
Beijing China
aonanli@bupt.edu.cn

Xiong Hugang
*State Grid Zhejiang electric power Co.,Ltd.Hangzhou power supply company*
Zhejiang China
344425874@qq.com

*Zhao Yongli
*Beijing University of Posts and Telecommunications*
Beijing China
yonglizhao@bupt.edu.cn



***Abstract*—This paper proposes a fast link recovery mechanism for high-concurrency services based on conflict avoidance in the fifth-generation fixed network(F5G). Results show that the mechanism improves the recovery rate and reduces delay in high-concurrency scenes.**



***Keywords—F5G, large scale optical network, signaling, fast link recovery***


## I. INTRODUCTION

After entering the F5G era, high resource utilization and efficient reliability have become important research objectives in optical networks. Due to the high speed and large bandwidth of the F5G, a large number of services will be transmitted in one link. If a link fault occurs in a large-scale optical network, it will cause many services to be disconnected, leading to a waste of network resources and increasing artificial maintenance cost. Compared with the F4G, due to the lower speed, the number of services sent at the same time will be much smaller than the F5G network. If a link fails, the number of lost services will be much lower than F5G. Therefore, optical network recovery techniques have become more important in F5G. However, the traditional link recovery technology is no longer applicable when it faces lots of recovery requests under high-concurrency service scenarios, which means considerable service recovery requests are initiated at the same time.

The traditional link recovery mechanism uses the link rerouting recovery mechanism to solve the problem of fast recovery of high-concurrency services. The mechanism is to use the shortest path at both ends of the faulty link to calculate a new section path, and then switch all services affected by the failure to the new section path. However, in the F5G network, a large number of services are already being transmitted in the new link. At this time, if the services on the faulty link are added to this new path, the total service bandwidth demand of this new path will be greater than the remaining bandwidth of the new path, which leads to a low recovery rate.

To improve the traditional link recovery mechanism, ZiJing Cheng proposes a Congestion-Aware Local Fast Recovery (CALFR) mechanism in 2017[1]. This method calculates multiple recovery paths at both ends of the faulty link and then switches them to different paths according to the bandwidth requirements of services affected by the fault, to achieve a high recovery rate. Y. Lin proposes a method: setting multiple protection links for the faulty link, which means as a fault occurs, the service on a fault link can be switched to a different protection link [2]. A proactive failure recovery algorithm, which was proposed by Ian Vilar Bastos, pre-computes disjoint primary and backup paths, and installs fast-failover groups in the primary path to reroute packets that encounter failures to the backup path [3]. Aiman Ghannami proposed a multipath routing with a fast failover scheme, which provides much resource utilization and fast recovery time in 2016 [4]. These solutions all achieve fault recovery by transferring the data of the faulty link to other paths. When a fault occurs, the services on the faulty link can be switched to different protection links. Forward link rerouting and backward link rerouting are proposed by Purnima [5]. These algorithms can reduce the number of signaling that needs to be issued when creating a new path. This method is different from transferring failed link data but improves the link recovery rate by reducing signaling overhead.

At present, there has been a lot of research on the recovery of faulty links, but in the face of large-scale and high-concurrency services scenarios, the existing methods are not suitable because the limited bandwidth of the new link, it is not possible to carry large amounts of services at the same time. Therefore, in this paper, an algorithm was proposed to cope with high-concurrency scenarios and solve the problems of low recovery rate and high distribution signaling delay.

## II. PROBLEM DESCRIPTION

The link recovery process includes link rerouting, signaling delivery, and node configuration. In the face of fast link recovery of high-concurrency services in the F5G, the main problem is a large number of services will accumulate on the same link due to link faults, resulting in low recovery rate and long recovery delay.

At present, the research on link recovery cannot realize fast link recovery of high-concurrency services. When the traditional link recovery mechanism faces high-concurrency service recovery requests, the recovery rate is too low due to insufficient resources. As shown in Fig. 1, there are four services $s_1$-$t_1$, $s_2$-$t_2$, $s_2$-$t_2$, and $s_3$-$t_3$ in the network, and the

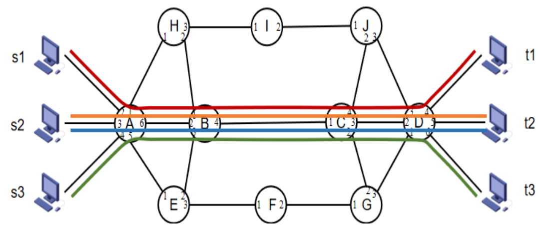


Fig. 1. Before link failure.

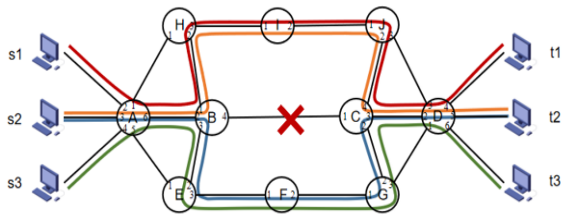

Fig. 2. Traditional link recovery mechanism.

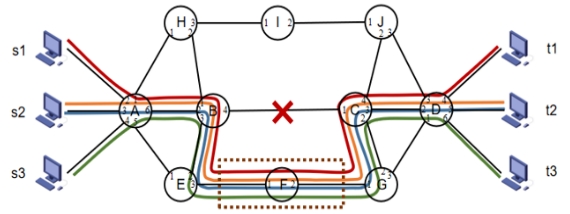

Fig. 3. Process of the CALFR.

contents in the figure represent the signaling messages of each service at each node. When a fault occurs between B-C, the four services in the network will be interrupted due to the fault.

As shown in Fig. 2, the traditional link recovery mechanism is to calculate a new path (B-E-F-G-C) between the faulty node B-C and then switch all services to the new path at node B. However, in the scenario of high-concurrency services, if all services are switched to the same path, there will be a problem of insufficient resources, resulting in an excessive blocking rate, and a low rate of service recovery. Therefore, the traditional link recovery mechanism cannot achieve effective recovery of high-concurrency services.

As shown in Fig. 3, when B-C fails, CALFR can calculate multiple rerouting paths between B-C, and switch them to different rerouting paths according to different service bandwidth requirements. But there are two problems with this scheme. As shown in Fig. 4(a), there is a service path A-E-B-C-D. As shown in Fig. 4(b), when B-C fails, the service recovery path is B-E-F-G-C, and its complete recovery path is

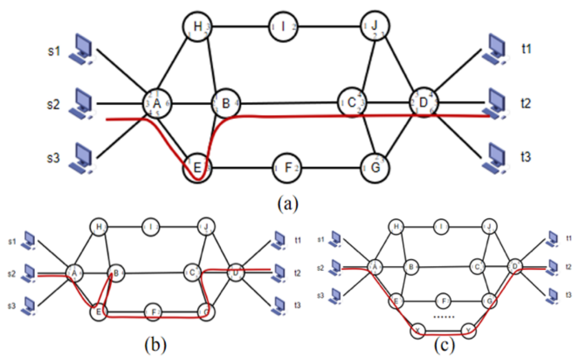

Fig. 4. Problems in CALFR.

Signaling Message in A

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 6 |
| 2 | $s_2$ | $t_2$ | output: 6 |
| 3 | $s_3$ | $t_3$ | output: 6 |
| 4 | $s_4$ | $t_4$ | output: 6 |

Signaling Message in B

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 4 |
| 2 | $s_2$ | $t_2$ | output: 4 |
| 3 | $s_3$ | $t_3$ | output: 4 |
| 4 | $s_4$ | $t_4$ | output: 4 |

Signaling Message in C

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 3 |
| 2 | $s_2$ | $t_2$ | output: 3 |
| 3 | $s_3$ | $t_3$ | output: 3 |
| 4 | $s_4$ | $t_4$ | output: 3 |

Signaling Message in D

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 4 |
| 2 | $s_2$ | $t_2$ | output: 5 |
| 3 | $s_3$ | $t_3$ | output: 5 |
| 4 | $s_4$ | $t_4$ | output: 6 |

Fig. 5. Signaling information of nodes before the failure.

Signaling Message in E

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 3 |
| 2 | $s_2$ | $t_2$ | output: 3 |
| 3 | $s_3$ | $t_3$ | output: 3 |
| 4 | $s_4$ | $t_4$ | output: 3 |

Signaling Message in G

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 2 |
| 2 | $s_2$ | $t_2$ | output: 2 |
| 3 | $s_3$ | $t_3$ | output: 2 |
| 4 | $s_4$ | $t_4$ | output: 2 |

Signaling Message in F

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 2 |
| 2 | $s_2$ | $t_2$ | output: 2 |
| 3 | $s_3$ | $t_3$ | output: 2 |
| 4 | $s_4$ | $t_4$ | output: 2 |

Signaling Message in B

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 3 |
| 2 | $s_2$ | $t_2$ | output: 3 |
| 3 | $s_3$ | $t_3$ | output: 3 |
| 4 | $s_4$ | $t_4$ | output: 3 |

Fig. 6. Signaling information of nodes in the traditional link.

A-B-E-F-G-C. The first problem existing in this solution is that loops may occur on the recovery path. As shown in Fig. 4(c), in the recovery process of this solution, to obtain more network resources, the recovery path will be too long. It can be seen that the second problem is that there may be detours in the recovery path. This solution can achieve a higher recovery rate by occupying more additional resources, but because the recovery path passes through the links connected to the nodes at both ends of the faulty link. Therefore, in the face of high-concurrency services recovery requests, a high bandwidth bottleneck will appear, which will lead to a sharp decrease in the recovery rate.

As researches study the recovery of high-concurrency service links, it is supposed to pay more attention to the issue of signaling delivery delay when creating a new recovery path besides studying the problem of recovery rate. Fig. 5 shows the signaling information of each node on the service path before the link fails. Fig. 6 and Fig. 7 respectively show the signaling information that needs to be delivered when the traditional link recovery mechanism and the congestion avoidance-based local recovery (CALR) mechanism establish the recovery path. The information of nodes A、C、D are not changed, so it is no longer marked in this figure. The information that can be obtained is: when the link recovery signaling is issued, the signaling information of different services on the same recovery path is the same. Therefore, when a recovery path is established, a large amount of repeated signaling information will be sent, resulting in an excessively long recovery time delay.

To sum up, the rapid link recovery of high-concurrency services mainly faces two problems: ①The rate of link recovery is too low; ②The signaling delivery delay is too long

Signaling Message in B

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 1 |
| 2 | $s_2$ | $t_2$ | output: 1 |
| 3 | $s_3$ | $t_3$ | output: 3 |
| 4 | $s_4$ | $t_4$ | output: 3 |

Signaling Message in E

| Flow id | src | dst | Actions |
|---|---|---|---|
| 3 | $s_3$ | $t_3$ | output: 3 |
| 4 | $s_4$ | $t_4$ | output: 3 |

Signaling Message in H

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 3 |
| 2 | $s_2$ | $t_2$ | output: 1 |

Signaling Message in F

| Flow id | src | dst | Actions |
|---|---|---|---|
| 3 | $s_3$ | $t_3$ | output: 2 |
| 4 | $s_4$ | $t_4$ | output: 2 |

Signaling Message in I

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 2 |
| 2 | $s_2$ | $t_2$ | output: 2 |

Signaling Message in G

| Flow id | src | dst | Actions |
|---|---|---|---|
| 3 | $s_3$ | $t_3$ | output: 2 |
| 4 | $s_4$ | $t_4$ | output: 2 |

Signaling Message in J

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output: 2 |
| 2 | $s_2$ | $t_2$ | output: 2 |

Fig. 7. Signaling information in CALFR.

when the link is restored. To solve these two problems, this chapter proposes a fast link recovery mechanism for high concurrent services based on collision avoidance. This scheme achieves a higher recovery rate by expanding the range of link recovery. At the same time, when signaling is delivered, its information of different services on the same recovery path is packaged to reduce the number of signaling required to be delivered, thereby reducing the signaling delivery delay.

## III. Fast Link Recovery Mechanism for High Concurrency Services based on Conflict Avoidance

### A. *Mathematical Model of the Mechanism*

When a failure occurs in the network, calculating one or more rerouted paths between the failed links will lead the recovery rate to be too low, and the recovery path will have loops and detours. Therefore, this paper proposes a fast link recovery mechanism for high concurrency services based on conflict avoidance (CA-HCSFLR). This scheme calculates multiple rerouted paths between the segment paths with a high coincidence degree according to the path coincidence of the failed service and then switches them to different paths according to the size of the service bandwidth requirements. At the same time, when the signaling is issued, the signaling information of the services with the same rerouting path can be packaged, which can effectively reduce the number of signaling to be issued, and thus reduce the service recovery delay. This chapter establishes the CA-HCSFLR mathematical model and obtains the most suitable scheme for the recovery of high-concurrency service links by solving the mathematical model. To achieve fast link recovery of high-concurrency services, our goal is to achieve a high rate of service recovery and reduce the number of signaling required to be issued.

**Notations:**

- *G (V, E, B)*: network topology, *V* denotes the set of nodes, *E* denotes the set of links, *B* denotes the set of link bandwidth.
- *l*: failed link in the network.
- *sou*: faulty link source node.
- *dis*: faulty link destination node.
- *n*: number of services affected by faults.
- *F*: services collection affected by faults.
- $B_i$: bandwidth requirements of the $i^{th}$ service affected by faults.
- $f_i$: the $i^{th}$ service affected by faults, $0 \leqq i \leqq m$.
- *S*: collection of segment paths containing failed link *l*.
- $s_i$: the $i^{th}$ segment path containing the failed link.
- $r_{ij}$: the $j^{th}$ rerouting path of $s_i$.
- $m_{ij}$: the number of nodes on $r_{ij}$.
- $N_{ij}$: collection of nodes on $r_{ij}$.
- $b_{ij}$: available bandwidth resources of $r_{ij}$.

**Variables:**

- $x_{ij}$: boolean variable that equals 1 if $r_{ij}$ is selected as the recovery path of $f_i$, and 0 otherwise.
- $x^p{}_q$: boolean variable that equals 1 if path *p* contains path *q*, and 0 otherwise.

**Objective:**

Maximum recovery rate:

$$max \frac{\sum_{j=1}^{m} x_{ij}}{n} \qquad (1)$$

Minimum number of signaling required to be issued:

$$min \sum_{i=1}^{n} \sum_{j=1}^{m} (x_{ij} + m_{ij} - 2) \qquad (2)$$

**Constraints:**

*1) Path constraint:*

The recovery path cannot contain a failed link, which needs to be satisfied $x^p{}_q = 0$.

*2) Bandwidth constraint:*

If $r_{ij}$ is selected as the recovery path of $f_i$, you need to be satisfied $b_{ij} > B_i$.

### B. *Algorithm Flow of the Mechanism*

When a fault occurs in the network, if one or more rerouting paths are calculated between the faulty links, the service blocking rate will get higher and cause loops and detours. Thus, this mechanism is based on the path overlap of the faulty services. It calculates multiple rerouting paths between section paths with a high degree of overlap and then switches them to different paths according to the size of the requirements of service bandwidth.

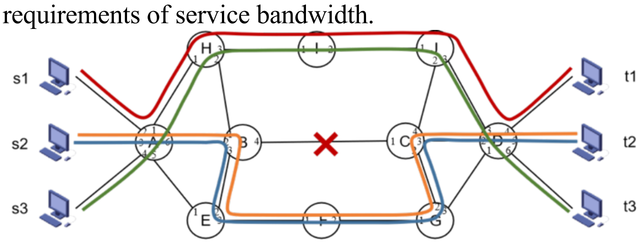


Fig. 8. CA-HCSFLR algorithm.

As shown in Fig. 8, two paths A-H-I-J-D and A-B-E-F-G-C-D are selected to calculate multiple reroute paths. Then, switching them to different paths according to the bandwidth requirements of the four services, which means that switching the two services s1-t1 and s3-t3 to the A-H-I-J-D path, and switching the two services of $s_2$-$t_2$ to the A-B-E-F-G-C-D path. This way can effectively avoid detours or loops caused by insufficient resources during service recovery. In the meantime, it can avoid some links that are prone to bandwidth bottlenecks in high-concurrency services.

Signaling Message in A

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output：1 |
| 2 | $s_2$ | $t_2$ | output：6 |
| 3 | $s_3$ | $t_3$ | output：6 |
| 4 | $s_4$ | $t_4$ | output：1 |

Signaling Message in D

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output：4 |
| 2 | $s_2$ | $t_2$ | output：5 |
| 2 | $s_2$ | $t_2$ | output：5 |
| 3 | $s_3$ | $t_3$ | output：6 |

Signaling Message in H

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output：3 |
| 2 | $s_2$ | $t_2$ | output：3 |

Signaling Message in I

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output：2 |
| 2 | $s_2$ | $t_2$ | output：2 |

Signaling Message in J

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output：3 |
| 2 | $s_2$ | $t_2$ | output：3 |

Signaling Message in F

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | output：2 |
| 3 | $s_3$ | $t_3$ | output：2 |

Signaling Message in B

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | output：3 |
| 3 | $s_3$ | $t_3$ | output：3 |

Signaling Message in C

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | output：3 |
| 3 | $s_3$ | $t_3$ | output：3 |

Signaling Message in E

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | output：2 |
| 3 | $s_3$ | $t_3$ | output：2 |

Signaling Message in G

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | output：5 |
| 3 | $s_3$ | $t_3$ | output：5 |

Fig. 9. Signaling information before packaging.

After calculating the rerouting path, it is necessary to send signaling to the rerouting path to establish the connection. Fig. 9 shows the signaling information that each node needs to deliver when using the CA-HCSFLR method to restore services. For the two services $s_1$-$t_1$ and $s_3$-$s_3$, signaling information needs to be sent to nodes A, H, I, J, and D. For the two services $s_2$-$t_2$, signaling needs to be sent to nodes B, E, F, G, C, and D. If the path of four services is re-established, it will need to deliver total 24 signaling information. It can be found that the signaling information of different services on the same path is identical, only the identification of the services is different.

Signaling Message in A

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | Set Agg Flow id 101， output： 1 |
| 4 | $s_3$ | $t_3$ | Set Agg Flow id 101， output： 1 |

Signaling Message in B

| Flow id | src | dst | Actions |
|---|---|---|---|
| 2 | $s_2$ | $t_2$ | Set Agg Flow id 102， output： 3 |
| 3 | $s_2$ | $t_2$ | Set Agg Flow id 102， output： 3 |

Signaling Message in E

| Aggregated Flow id | Actions |
|---|---|
| 102 | output： 3 |

Signaling Message in D

| Flow id | src | dst | Actions |
|---|---|---|---|
| 1 | $s_1$ | $t_1$ | output： 4 |
| 2 | $s_2$ | $t_2$ | output： 5 |
| 3 | $s_3$ | $t_3$ | output： 5 |
| 4 | $s_4$ | $t_4$ | output： 6 |

Signaling Message in F

| Aggregated Flow id | Actions |
|---|---|
| 102 | output： 2 |

Signaling Message in G

| Aggregated Flow id | Actions |
|---|---|
| 102 | output： 2 |

Signaling Message in C

| Aggregated Flow id | Actions |
|---|---|
| 102 | output： 3 |

Signaling Message in H

| Aggregated Flow id | Actions |
|---|---|
| 101 | output： 1 |

Signaling Message in I

| Aggregated Flow id | Actions |
|---|---|
| 101 | output： 2 |

Signaling Message in J

| Aggregated Flow id | Actions |
|---|---|
| 101 | output： 3 |

Fig. 10. Signaling information after packaging.

Fig. 10 lists the signaling information delivered by each node after packaging. For the two services $s_1$-$t_1$ and $s_3$-$s_3$, the signaling information can be combined at node A, and the two services can be represented by the same identifier (101). For the two services of $s_2$-$t_2$, the signaling information can be combined at Node B, and the two services can be represented by the same identifier (102). After signaling merging, a total of 15 pieces of signaling information need to be delivered to complete the establishment of these four service paths, which is 7 pieces of signaling information less than before packaging. Table 1 shows the local rerouting algorithm flow based on conflict avoidance, and the specific steps are as follows:

Table 1 TRAFFIC AGGREGATION LOCAL REROUTE ALGORITHM

**Algorithm:** Fast Link Recovery Mechanism for High Concurrency Services based on Conflict Avoidance Algorithm

**Input: G= (V, E), failed link *l***

**Output: All the disrupted traffic flows are restored**

1: The SDN controller monitors the status of the network topology, and finds link ***l*** is disrupted;

2: $\boldsymbol{G} \leftarrow \boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E} - \boldsymbol{l})$

3: Create set ***M*** ← all the traffic flows traversing link ***l***;

4: Sort ***M*** in a descending order of the bandwidth of the traffic flow;

5: Set a max hops threshold ***h***;

6: Create set ***N*** ← all segments of ***M*** whose hops are less than ***h*** and contains ***l***;

7: Compute multiple reroute paths $\boldsymbol{R_i}$ of $\boldsymbol{n_i}$;

8: Sort $\boldsymbol{R_i}$ in an ascending order of the path hops;

9: **for each** $\boldsymbol{m_i} \in \boldsymbol{M}$ **do**

10: Create empty set ***U***;

11: **if** $\boldsymbol{m_i}$ contains $\boldsymbol{n_i}$ **then**

12: **for each** $\boldsymbol{r_{ij}} \in \boldsymbol{R_i}$ **do**

13: **if** $\boldsymbol{b_{ij}} > \boldsymbol{a_i}$ **then**

14: $\boldsymbol{U} \cup \boldsymbol{r_{ij}}$;

15: $\boldsymbol{b_{ij}} = \boldsymbol{b_{ij}} - \boldsymbol{a_i}$;

16: **break;**

17: **else** continue**;**

18: **else** continue**;**

19: **end**;

## IV. SIMULATION RESULTS

In order to evaluate the performance of CA-HCSFLR, two experiments are conducted in this paper. Simulation 1 and 2 respectively verified the improvement of the rate of service recovery and the delay in sending service recovery.

### A. Simulation conditions

Simulation 1 is mainly used to evaluate the performance of CA-HCSFLR in terms of service recovery rate, resource utilization rate, and average hops of the recovery path. The comparison algorithm is CALFR and K-Shortest Path (KSP) algorithm (K=3). Python is used as a programming language and the USNET topology (24 nodes, 43 links) shown in Fig. 11. The bandwidth of each link is 10G. The simulation services model is high concurrency. The bandwidth of each service is randomly selected between 2.4M and 10M, and the number of services is between 500 and 1700.

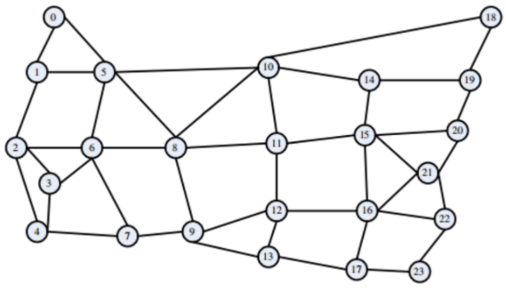

Fig. 11. USNET topology.

Simulation 2 is mainly used to evaluate the performance of CA-HCSFLR in terms of recovery delay and the number of required signaling. CALFR, KSP, CA-HCSFLR, link recovery mechanism for high concurrency services based on conflict avoidance (CA-HCSLR), and CALR were selected for the comparison. NS-3 network is used as simulation software and the topology of 1132 nodes (1132 nodes, 1320 links) shown in Fig. 12. The controller is located in the center of the topology and connected to the eight core nodes (orange nodes) in the center. The bandwidth of the link connected to the controller is set to 10M, and the bandwidth of other links is 5M. The services model is high concurrency, which means signaling for high concurrency services needs to be issued at the same time and the number of services is between 1,000 and 2,000.

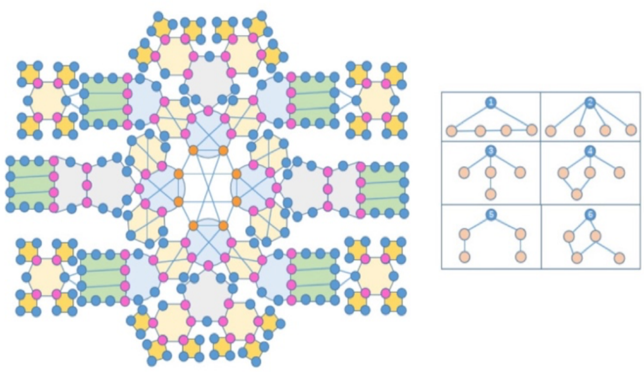

Fig. 12. 1132 nodes topology.

## B. Simulation results analysis

Fig. 13 compares the service recovery rates of CA-HCSFLR, CALR, and KSP in the USNET topology. The simulation results show that when the number of services is between 500 and 1100, the rate of service recovery: CA-HCSFLR>CALFR>KSP; when the number of services is between 1400 and 1700, the rate of service recovery: CA-HCSFLR>KSP>CALFR. It can be seen that the proposed CA-HCSFLR scheme can achieve a high recovery rate under high-concurrency services. This is because the CA-HCSFLR expands the scope of link recovery and calculates a large number of recovery paths between different section paths including faulty links, so as to obtain more network resources and improve the rate of service recovery. In the CALFR, when the number of services increases to 1700, the recovery rate drops 52%. Because the resource consumption of the links connected to the nodes at both ends of the faulty link is too large, which will lead to a large bandwidth bottleneck so that the services can be recovered when the number of services is small but cannot be recovered in the case of a large number of services. And there is no obvious decline in KSP.

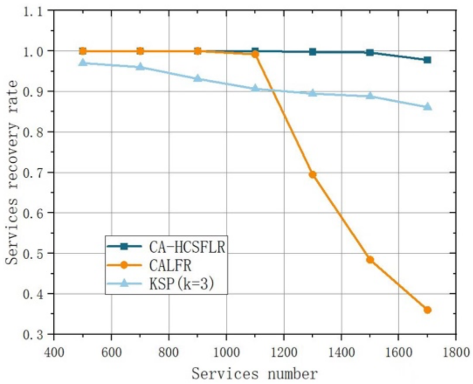


Fig. 13. Services recovery success rate.

Fig. 14 compares the resource utilization of the three mechanisms. The simulation results show that when the number of services is between 500 and 1100, the resource utilization rate is: CALFR>CA-HCSFLR>KSP. When the number of services is between 1400 and 1700, resource utilization: CA-HCSFLR>KSP>CALFR. When the number of services is large, the resource utilization rate of the CA-HCSFLR is the highest. This result verifies that the CA-HCSFLR scheme can obtain more network resources to improve the recovery rate. The resource utilization rate of the CALFR solution drops almost 47% when the number of services is 1700, and the reason is consistent with the above-mentioned reason for the sudden drop in the recovery rate.

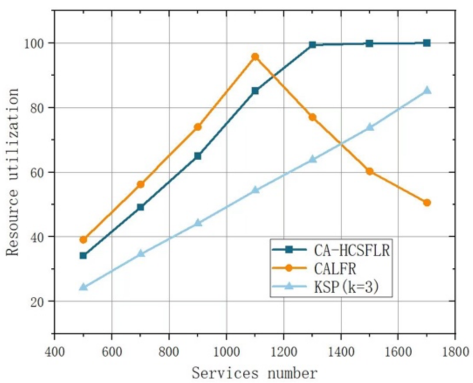


Fig. 14. Resource utilization.

Fig. 15 compares the average hop counts of the recovery paths of the three solutions. Simulation results show that the average number of hops in the recovery path: CALFR>CA-HCSFLR>KSP. It can be seen that CA-HCSFLR effectively avoids the detour problem and achieves a better recovery effect. In summary, CA-HCSFLR can change the rerouting only between faulty links to section paths containing faulty links, and obtain more network resources by expanding the recovery range. At the same time, it can effectively avoid bandwidth bottlenecks and detours during link recovery. Besides, it can achieve a higher recovery rate and better recovery effect compared with traditional methods.

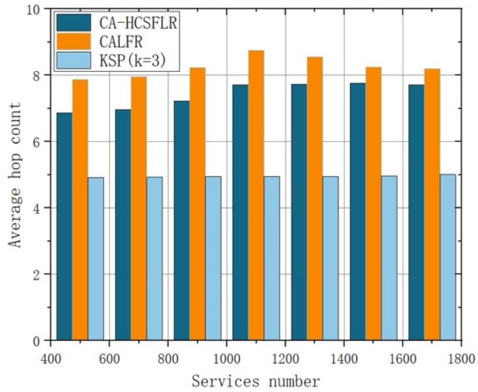


Fig. 15. Average length of recovery path.

Figs. 16 and 17 respectively compares the recovery delay of high-concurrency services the number of signaling that needs to be delivered. The recovery delay mainly includes the delay of rerouting and signaling delivery.

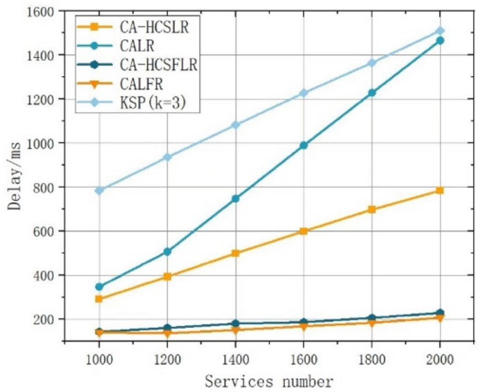


Fig. 16. Comparison of recovery delay of each scheme.

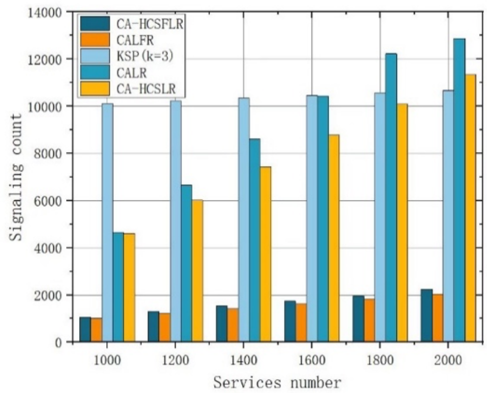


Fig. 17. Comparison of the number of signaling issued by each scheme.

Comparison schemes include CA-HCSFLR, CALR, KSP, CA-HCSLR, and CALR. Simulation results show that both the recovery delay and the number of signaling which required to be delivered should satisfy: KSP>CALR>CA-HCSLR>CA-HCSFLR>CAFLR. It can be seen that CA-HCSFLR and CAFLR can achieve a smaller link recovery delay. Because it will assign the same identifier to the services of the same recovery path during service recovery, to realize the combination of signaling information of different services on the same path, can effectively reduce the number and the recovery delay. At the same time, the recovery delay of CAFLR is shorter than CA-HCSFLR, because in the CA-HCSFLR scheme, more signaling information can be combined and the recovery delay can be reduced effectively.

## V. Conclusion

In this paper, the proposed mechanism solves the problem of fast recovery of high-concurrency services from the aspect of fast link recovery of high-concurrency services in the F5G network. Given the limitations of existing solutions, this paper proposes a fast link recovery mechanism for high-concurrency services based on conflict avoidance. Simulation 1 and 2 verify the recovery rate and recovery speed of the proposed scheme. From the above simulation results, it can be seen that CA-HCSFLR could achieve a high rate of recovery while occupying fewer additional resources and realize fast link recovery of high-concurrency services through signaling packaging.

## Acknowledgment

This work has been supported in part by China State Grid Corp Science and Technology Project (5700-202219205A-1-1-ZN).

## References

[1] Cheng Z, Zhang X, Li Y, et al. Congestion-aware local reroute for fast failure recovery in software-defined networks[J]. Journal of Optical Communications and Networking, 2017, 9(11): 934-944.

[2] Y. Lin, H. Teng, C. Hsu, C. Liao, and Y. Lai, "Fast failover and switchover for link failures and congestion in software defined networks," in IEEE Int. Conf. on Communications (ICC), 2016.

[3] I. V. Bastos, V. C. Ferreira, D. C. Muchaluat-Saade, C. V. N. de Albuquerque and I. M. Moraes, "Path Recovery Algorithm Using Fast-Failover for Software-Defined Networks," 2020 4th Conference on Cloud and Internet of Things (CIoT), Niteroi, Brazil, 2020, pp. 49-52, doi: 10.1109/CIoT50422.2020.9244292.

[4] A iman Ghannami and ChenXi Shao, "Efficient fast recovery mechanism in Software-Defined Networks: Multipath routing approach," 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST), Barcelona, Spain, 2016, pp. 432-435, doi: 10.1109/ICITST.2016.7856747.

[5] M ohan P M, Truong-Huu T, Gurusamy M. TCAM-aware local rerouting for fast and efficient failure recovery in software defined networks[C]//2015 IEEE Global Communications Conference (GLOBECOM). IEEE, 2015: 1-6.