ISBN: 978-981-18-5852-9
WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications
doi: 10.18178/wcse.2022.04.075

Network Slicing Resource Mapping Algorithm Oriented to Delay and Reliability Awareness

Ji-Hong Zhao [1], Zhi-Lin Lei [1], Hua Qu [2] and Hang Song [1]

[1] Xi’an University of Post & Telecommunications School of Communication and Information Engineering

[2] Xi’an Jiaotong University School of Electronic and Information Engineering

Abstract. Network slicing is an important network technology that meets different 5G application scenarios. Aiming at the differentiated needs of business and the problem of providing customized network services for tenants through resource mapping, a network slicing resource mapping algorithm oriented to latency and reliability awareness is proposed. First, in order to fully perceive the latency and reliability indicators of the underlying nodes, the reinforcement learning model is introduced into the node mapping process, and the ability of the model to interact with the environment is used to enable it to perceive changes in the underlying state and make dynamic decisions. Secondly, in the link mapping stage, facing the different requirements of delay and reliability, a hierarchical mapping strategy is adopted to realize a network slice resource mapping algorithm for differentiated services. Finally, through simulation and comparison with other algorithms, the results show that the proposed algorithm not only meets the requirements of reliability and delay, but also improves the request acceptance rate and resource utilization rate, and guarantees the delay and reliability of service quality.

Keywords: Network slicing, Business-oriented, Resource allocation, reinforcement learning.

1. Introduction

As a key network technology to meet the diverse requirements of 5G, network slicing is essentially a customized virtual network that shares infrastructure resources but is logically completely isolated. Network slice resource mapping is a key step in network slice generation [1,2]. By abstracting the user's business requirements into network resource requirements, network slices can be allocated on demand and isolated from each other.

With the rise of artificial intelligence (AI) field, many researchers have begun to try to integrate AI into network slicing technology. In the research of network slice resource mapping, Li R et al. proposed to apply deep reinforcement learning to network slice mapping problem [3], which is considered to be a promising research direction. H. Yao, B. Zhang [4] and others used the reinforcement learning (RL) model to integrate with the VNE algorithm to achieve dynamic virtual network mapping, and the results were better than other algorithms in terms of long-term income and acceptance rate. Wang C, Zheng F [5] and others added a neural network to the algorithm, and designed a reward function according to the node and link resource consumption, make a connection between the node mapping and link mapping stages of VNE. In the literature [6], the author uses the RL model to introduce the differential characteristics of nodes and links into the algorithm, and simultaneously considers the node and link mapping, and the results are better in the long-term benefit-cost ratio and acceptance rate. RL has the ability to frequently interact with the environment, and can fully perceive the dynamic changes of the underlying network resources. On the other hand, with the help of its reward function system, slicing can perform more accurate mapping when facing diversified business requirements.

The International Telecommunication Union (ITU) sum-marized the three main application scenarios of 5G networks [7], the Ultra Reliable and Low Latency Communication (uRLLC) scenario requires the network to have the characteristics of high reliability and low latency. First, in the existing delay-aware research, the authors of [8-10] assign the delay as a resource to the underlying link, and set the algorithm optimization goal to minimize link delay resources to meet low-latency requirements slicing business request. The authors of the literature [11,12] transformed the optimization goal of minimizing the delay into the topology goal of the minimum number of hops in the path, and achieved the purpose of reducing the delay by reducing the node size and the number of transmission hops in the path. Secondly, in the current research on reliability perception, the literature [13-17] improves the reliability of the network by setting backup. The network performance can be recovered faster by means of backup, but the backup will occupy more resources and increase the mapping cost. In [18,19], the author quantifies the failure of the underlying equipment as a reliability evaluation index, and proposes a slice mapping algorithm for different business types, which not only ensures the reliability of slices, but also meets the diversified needs of the business. In [20], the author comprehensively considers the underlying topology, reliability evaluation of a single node and the reliability evaluation of the adjacent environment of the node as the reliability indicators of the node. Through multi-dimensional resource evaluation, the underlying failure probability of the slice is reduced, and the reliability of the slice is guaranteed. Therefore, in the face of the problem of diversified business needs, the corresponding indicators of delay and reliability also have diversified needs. A single perception indicator may be one-sided, resulting in insufficient perception of the underlying resources, resulting in sub-optimal mapping results.

In summary, a single resource-aware method cannot effectively deal with the multi-demand problem of network slicing service types. At the same time, due to the limited physical resources and the dynamic nature of business requests, unreasonable mapping will produce sub-optimal mapping results and reduce system performance and user experience. Therefore, there is an urgent need for efficient and novel intelligence that can sense environmental changes and adjust policies in a timely manner mapping algorithm. In view of the above problems, this paper proposes a network slice resource mapping algorithm based on reinforcement learning for latency and reliability perception to achieve accurate perception of latency and reliability. Considering the topology parameters and characteristic parameters comprehensively, the network slice requests are classified and processed to realize the network slice resource mapping of differentiated services.

2. System Model and Problem Statement

2.1. Low-level Network Model

This article abstractly defines the physical Substrate Network (SN) as a weighted undirected graph, $G_S=(N_S,L_S,A_S^N,A_S^L)$. Where N_S represents a collection of physical nodes of the underlying network, which is composed of general physical servers; A_S^N represents the node attributes of the physical server, such as CPU resources and reliability evaluation; L_S represents the set of physical links connecting the server, and A_S^L represents the attributes of the physical link, such as bandwidth resources and delay evaluation.

2.2. Slice Request Model

The Slice Request Model will be represented by $G_V=(N_V,L_V,R_V^N,R_V^L,R_V^Q)$, Where N_V is the set of all nodes in the slicing request, and R_V^N is the resource constraint of each node in the slicing request, such as the CPU capacity requirement of the node; L_V represents the set of all links in the slicing request, R_V^L represents the resource constraint of each link in the slicing request, such as the bandwidth requirement of the link; R_V^Q represents the overall evaluation constraint of the current slicing request, such as reliability evaluation RD and delay evaluation DL, which are mainly studied in this paper.

2.3. Slice Mapping Problem Statement

The process of network slice resource mapping starts with a network slice request initiated by a tenant, and physical network resources are allocated to the target slice according to the resource requirements of the current request. The mapping process from the slice request G_V to the underlying network G_S is called the resource mapping process of the network slice G_S^V. It is represented by M: $G_v^k \rightarrow G_s^k=(\mathrm{N}_s^k,\mathrm{P}_s^k)$. Where $\mathrm{N}_s^k \in N_S$ represents the set of underlying nodes to be mapped by virtual node N_v^k in the kth request, the underlying node in the set allocates resources for the virtual node and meets its resource requirements; $P_s^k \in P_S$ represents the virtual link E to be mapped in the kth request The set of underlying paths in the set, where the underlying path allocates resources for the virtual link and meets its resource requirements, where P_S represents the set of reachable paths between pairs of nodes in the underlying physical network.

3. Algorithm Design

The network slicing resource mapping algorithm (OLR-NS) proposed in this paper for latency and reliability awareness uses two-stage mapping, which is divided into a node mapping phase and a link mapping phase. In the node mapping stage, the reinforcement learning model is introduced, and the strategy gradient method is used to train the model. Among them, reliability is the attribute of the node, and delay is the attribute of the link. For the problem of delay awareness, the formula is used to convert to the node attribute, which is added to the state matrix of the node. Through the improved node state matrix, the model can fully perceive the delay and reliability of the underlying nodes. In the link mapping stage, the reliability perception is converted from node attributes to link attributes through formulas to strengthen the delay and reliability perception in the link. At the same time, a hierarchical mapping strategy is adopted according to different slicing requirements.

The algorithm flow is shown in Fig. 1.

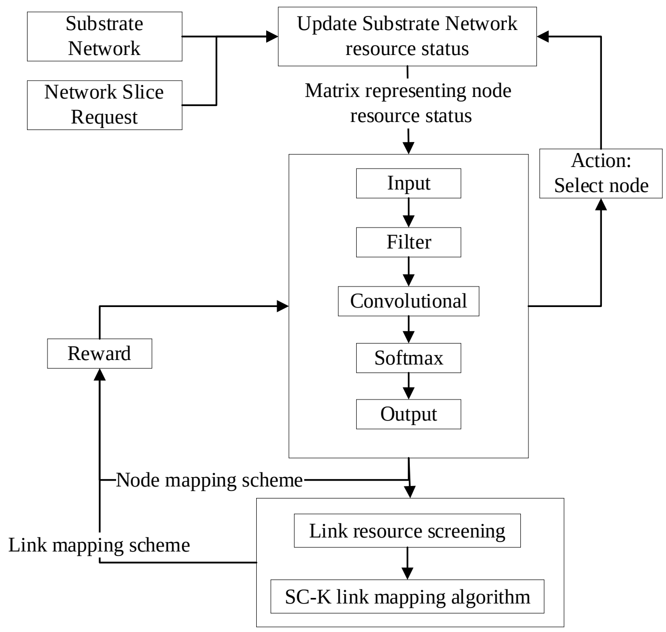

Fig. 1: System model.

There are four steps: First, input the slicing request and the underlying network data, update the node state matrix of the underlying network, and enter the mapping strategy network. Second, after the node mapping strategy network obtains the state matrix, the mapping probability of each node is calculated, and the node with the highest probability is selected as the action of the strategy network. Until the mapping of all nodes requested this time is completed, the second step is repeated. Third, the node mapping scheme will be passed into the link mapping stage. Then, the bottom-layer path in the bottom-layer network that does not meet the resource constraints of the link mapping is cleaned. The cleaned bottom layer network topology is sent to the SC-K link mapping algorithm, and the link mapping result of this request is obtained. Fourth, get the node mapping scheme and link mapping scheme of the request, calculate the reward function and feed it back to the policy network for iterative update.

3.1. Node Mapping

In the node mapping stage, the node mapping is modeled as a Markov decision process, and a reinforcement learning model is introduced to train the mapping strategy network. Through training to learn the optimal node mapping strategy, the Markov decision process includes four basic elements: state, action, strategy, and reward. Among them, the node feature state matrix extracted by the underlying network is regarded as the state in the decision-making process. It is the mapping between the current virtual node and the selected physical node; the strategy is the mapping probability distribution given by the current state of the underlying network. The reward is the feedback after the environment gives an action based on the strategy.

Environmental Status: Whether reinforcement learning can complete the mapping efficiently, the agent can fully perceive the underlying resources is the key. The node feature vector used to describe the state of the node is expressed as: S^k =[$Res(n_s^k)$, $BW(n_s^k)$, $DEG(n_k^s)$, $CL(n_s^k)$, $DS(n_s^k)$, $RD(n_s^k)$, $DL(n_s^k)$]. They are in order: node capacity, link bandwidth, local topology features-degree centrality, two global topology features-proximity centrality and node aggregation capability, two business-oriented node feature vectors delay evaluation and reliability Sexual evaluation. Part of it is shown as follows.

a. Selection Node Aggregation Capability: measure the balance between the node and the mapped node Both distance, as in (1). Where Molecular part represents the physical node where the virtual node that has been mapped in a slicing request is located.

$$DS(n_s^k) = \frac{\sum_{\bar{n}_s \in \bar{N}_s} SP(n_s^k, \bar{n}_s)}{\left|\bar{N}_s\right| + 1} \tag{1}$$

Reliability Evaluation Function: Measure the proportion of the non-failure time of the node server in the total running time, as in (2). Where $MTBF(n_s^k)$ represents the mean time between server failures of the underlying device node k, and $MTTR(n_s^k)$ represents the server failure repair time of the underlying device node k.

$$RD(n_s^k) = \frac{MTBF(n_s^k)}{MTBF(n_s^k) + MTTR(n_s^k)} \tag{2}$$

b. Delay Evaluation: Measure the average value of the delay evaluation from this node to other nodes in the network, as in (3). Where $sp(n_s^k, n_s^m)$ is represented as the shortest path between two nodes.

$$DL(n_s^k) = \frac{\sum_{m=i}^{N} \sum_{l \in sp(n_s^k, n_s^m)} D(l)}{N-1} \tag{3}$$

(1) Node Mapping Strategy Network: The node strategy network model includes: input layer, convolution calculation layer, node screening, probability calculation layer, and output layer. The strategy network takes the node state matrix of the current underlying network as input. After several layers of network conversion, the probability of each node being selected is finally obtained, and the output is a mappable probability distribution.

In the convolutional calculation layer, a weighted calculation is performed on the node state matrix, and the resource evaluation vector of the bottom node is calculated and generated, as in (4). And it is used to evaluate the matching degree of the underlying node to the virtual node of the current request.

$$\beta^k = \begin{cases} \omega \cdot S^k + b \ , & if \ \ \omega \cdot S^k + b \geq 0 \\ 0 \ , & otherwise \end{cases} \tag{4}$$

Where β^k represents the k-th output of the convolution, ω represents the weight vector, and b represents the bias vector.

In the probability calculation layer, the resource evaluation vector of the selected node is converted into the probability of the node being selected. The higher the evaluation, the higher the matching degree between the node and the current request mapping node, and the greater the corresponding probability. Finally, the output is a set of probability distributions of each node, as in (5).

$$\eta^k = \frac{e^{\beta^k}}{\sum_i e^{\beta^i}} \tag{5}$$

(2) Reward Function: The reward function affects the iterative direction of the agent's learning strategy. It is the criterion for the agent to select actions through the strategy. Therefore, in order to enable the model to generate the optimal mapping result that satisfies the slice request, make the result satisfy the maximum tolerance values for latency and reliability in the slicing request. In this paper, the evaluation function of service quality loss is added to the reward function. At the same time, the reward function of the reinforcement learning model is redefined in the case of collaboratively considering the resource utilization, as in (6) and (7).

$$uti = NodeRU * LinkRU \tag{6}$$

$$R = \begin{cases} uti - QS & , \ if \ map \ is \ successful \\ 0 & , \ otherwise \end{cases} \tag{7}$$

(3) Gradient Update: In this paper, a small batch gradient descent method is used to dynamically update the network parameters of the strategy. First, use cross entropy to define the loss function of the node mapping stage, as in (8). Where θ_k is a vector coded by $one-hot$, representing the situation of the mapping node.

$$L(\theta,\delta) = -\sum_k \theta_k \log(\delta_k) \tag{8}$$

In the iterative process, the number of $batch\_size$ samples is selected for one update, and the parameter σ is introduced to coordinate the size of the gradient and the calculation speed of training. If σ is too large, the output may be unstable and unable to converge during training; if σ is too small, the training process will become extremely slow. In this article, the parameters $batch\_size$ and σ are set to 0.05 and 100 after many times of tuning. The process of the node mapping algorithm is as follows:

Algorithm 1 Node Strategy Network Training Model

Require: Substrate network G_s; Network slicing request G_V; Iterations number p

Ensure: Strategic network model $Model$; Weight vector ω; Bias vector b

initialize(ω , b)
while iteration < p **do**
  counter=0
  for $req \in G_V$ **do**
    for $n_v \in req$ **do**
      initialize(M_{sub})
      for $n_s^k \in G_S$ **do**
        $M_{sub} \leftarrow M_{sub} + S^k$
      end for
      $\alpha'(M_{sub})$ //Feature matrix normalization
      $\eta^k \leftarrow police\_network \ with \ \omega \ and \ b$
      n_s = e-greedy_select(η^k)
      $update \ resource \ in \ G_S$
    end for
    if $\forall n_v \in req \ is \ Mapped$ **then**
      $sc-k\_linkmap(req)$
    end if
    if $req\_Map \ is \ successfully$ **then**
      $reward = r(req) = uti - loss$
      $computeGradient(reward)$
    end if
    counter++
    if counter = $batch\_size$ **then**
      $update\ (\sigma)$ //Update learning rate
    end if
  end for
  iteration++
end while

3.2. Link Mapping

This paper proposes a KSP-based service customization link mapping algorithm (SC-K) to implement the link mapping phase of each request. First, before link mapping, the underlying links that do not meet the current link bandwidth requirements are temporarily erased. Thus, a new underlying network to ensure the resource requirements of the virtual link. Secondly, the delay constraint is compared with the delay requirement in the slicing request in the form of the cumulative sum of the underlying link delays, which can not only ensure the overall delay profile of the slicing request, but also restrict the number of hops of the path. Function, indirectly reduce the bandwidth consumption of the link. The reliability evaluation compares the product of each node in the underlying path with the reliability requirements in the slicing request, and combines the characteristics of the node with the link mapping to strengthen the relevance of the two-stage mapping of the node and the link.

In order to meet diversified service requirements, this article classifies the slice requests:

- Type A service, which meets the strict requirements of delay and reliability, but does not require high bandwidth. Its one-way delay does not exceed 50ms, the stability evaluation requirement is not less than 98%, and the bandwidth demand only accounts for 0.5Mbps;
- Type B services, compared to Type A, will have less delay and reliability requirements, but have a certain bandwidth requirement. Its one-way delay is not more than 100ms, the stability evaluation requirement is not less than 97%, and the bandwidth requirement is 2.5Mbps;
- Type C services are not sensitive to delay and reliability, so there is no specific tolerance value. The bandwidth occupancy demand is high. Based on the above data, the final results are shown in Table 1.

Table 1: Categories for Slicing Requests

Domain Name	Bandwidth Resources	Reliability Evaluation	Latency Evaluation
A	0-1Mbps	97-99%	50ms
B	2-4Mbps	95-97%	100ms
C	1-5Mbps	-	-

Algorithm 2 Link Mapping Algorithm

Require: Substrate network G_s **; Network slicing request** G_V **; Node mapping result** $nodemap$

Ensure: Link mapping result $linkmap$

$linkmap \leftarrow \varnothing$
initialize ($linkmap$ **)**
for virtual link l_v **in** G_V **do**
    $G_S^{temp} \leftarrow G_S$
    for substrate link l_s **in** G_S **do**
        if $B(l_s) < B(l_v)$ **then**
            cut l_s **in** G_S^{temp}
        end if
    end for
    Node A= $nodemap$ **.get(.startNode)**
    Node B= $nodemap$ **.get(.endNode)**
    Node Pair = < Node A , Node B >
    $PATH$ **= KSP(Node Pair)**
    for path p **in** $PATH$ **do**
        Calculate the underlying network delay sdl **and reliability** srd
        Obtain virtual request latency vdl **and reliability** vrd
        if $sdl \le vdl$ **,** $srd \ge vrd$ **then**
            $linkmap \leftarrow linkmap + \{l_v \rightarrow p\}$
            break
        end if
    end for
end for
return $linkmap$

3.3. OLR-NS Online Mapping Algorithm

In general, the OLR-NS algorithm first trains the node mapping strategy network through the reinforcement learning model in the existing network environment, completes the node mapping stage through the strategy network, and then transmits the node mapping result to the link mapping algorithm $sc-k$. Finally it realizes global online mapping.

4. Experimental Simulation and Result Analysis

4.1. Experimental Environment and Simulation Method

In the simulation experiment, the underlying network topology and network slicing request topology used in this article are both generated using GT-ITM tools, and the locations of all physical nodes in the underlying network are random, with a probability of 0.5 between nodes connect. The experiment is simulated by using the TensorFlow reinforcement learning framework in python in the anaconda environment. The specific parameters are shown in Table II and Table III.

Table 1: Substrate Network Parameter

Parameter Name	Value Range
Number of nodes	100
Number of links	570
Node CPU resources	50-100
Link bandwidth resources	50-100 MHZ
Node reliability	0.95-0.99
Link delay	0-50

Table 2: Network Slice Request Parameter

Parameter name	Value range
Running time	50000
Number of slice requests	2000
Inter-arrival	Poisson distribution, reaching 40 requests per 1000 time units on average
Life cycle time	index distribution
Number of nodes	2-10

Node CPU resources	0-50
Link bandwidth resources	0-50 MHZ

4.2. Simulation Result Analysis

(1) Evaluation function: The evaluation indicators in this article mainly include: slice successful access rate, long-term average benefit-cost ratio, underlying resource node utilization, link utilization, average slice deployment time (algorithm time complexity), and service quality loss.

a. Request Access Rate: The ratio of the number of successful mapping requests to the number of request arrivals, as in (9). Where $\sum_{t=0}^{T} M_s(G_v)$ represents the number of slice requests successfully deployed in a period of time, and $\sum_{t=0}^{T} G_v$ represents the total number of VNRs that have arrived in this period of time.

$$requestAR = \frac{\sum_{t=0}^{T} M_s(G_v)}{\sum_{t=0}^{T} G_v} \tag{9}$$

b. Long-term Average Revenue to Cost Ratio: First, after a request is successfully deployed at time T, its revenue is defined as in (10), and its cost is defined as in (11). Where $cpu(n_v)$ represents the CPU resource constraint required by the virtual node n_v in the request, $bw(l_v)$ represents the bandwidth resource constraint required by the virtual link l_v in the request, and $M(l_v)$ represents the underlying physical path that maps the virtual link. Then, the long-term average benefit-cost ratio is defined as in (12).

$$RE(G_v,t) = \sum_{n_v \in N_v} cpu(n_v) + \sum_{l_v \in L_v} bw(l_v) \tag{10}$$

$$CO(G_v,t) = \sum_{n_v \in N_v} cpu(n_v) + \sum_{l_v \in L_v} \sum_{l_s \in M(l_v)} bw(l_v) \tag{11}$$

$$RC(G_v,t) = \frac{\sum_{t=0}^{T} \sum_{G_v \in VN_{map}(t)} RE(G_v,t)}{\sum_{t=0}^{T} \sum_{G_v \in VN_{map}(t)} CO(G_v,t)} \tag{12}$$

c. Resource Utilization: During system operation, the average value of the ratio of occupied resources to global resources, as in Long-term average node resource utilization (13) and Long-term average link resource utilization (14). Where $S_c(n)$ and $C(n)$ respectively represent the remaining available CPU resources and total CPU resources of the current node n, $|N_s|$ represents the number of physical nodes, and V represents the total number of requests that arrived in the time period; $S_b(l)$ and $B(l)$ respectively represent the remaining available bandwidth of the current link Resources and total bandwidth resources, $|L_s|$ represents the number of physical links, and V represents the total number of requests that arrive in this time period.

$$NodeRU = \sum_{t=0}^{T} \frac{\sum_{n \in N_s} 1 - \frac{S_c(n)}{C(n)}}{|N_s| V} \tag{13}$$

$$LinkRU = \sum_{t=0}^{T} \frac{\sum_{l \in L_s} 1 - \frac{S_b(l)}{B(l)}}{|L_s| V} \tag{14}$$

d. Service Quality Loss: Quantify the network service experience of the user side, and express the difference between the reliability and delay that the user actually experienced and the request after the mapping is completed by the loss of network service quality, as in (15). Where R_s and D_s respectively represent the actual reliability and delay after the current slicing request is mapped to the underlying network, and R_v and D_v respectively represent the reliability and delay tolerance thresholds of the current slicing request.

$$QS = \frac{R_s - R_v}{R_v} + \frac{D_s - D_v}{D_v} \tag{15}$$

(2) Analysis of results: This article will use the evaluation indicators introduced in the previous section to evaluate the performance of the OLR-NS implementation algorithm, and combine the deployment algorithm of this article with the distributed mapping algorithm (DC) based on the greedy strategy [21] and the Markov random walk based Topology-aware mapping algorithms (MCRM) [22] are compared. Both the above two algorithms and the algorithm proposed in this paper consider the delay requirements and reliability requirements of the slice request. This article will evaluate the performance of the algorithm from the aspects of slice access rate, resource utilization, mapping cost ratio, service quality loss, and average deployment time of requests, and verify the effectiveness of the slicing deployment method in ultra-reliable and low-latency business scenarios.

a. Algorithm Runtime Analysis: First, in Table IV, the time consumption of the three algorithms is shown, where the online mapping time is taken from the average of ten runs of the algorithm. It can be seen from the table that the online mapping time of the algorithm proposed in this paper is at a lower position than the comparison algorithm. This is due to the ability of the reinforcement learning model. After a long time offline training, it can take a small time cost to complete the user The online mapping section of the request.

Table 1: Average Algorithm Runtime

Algorithm Name	Online Mapping Time (Seconds)	Offline Training Time (Seconds)
OLR-NS	253.40786480903625	799.9914360046387
DC	74.97001695632935	-
MCRM	4209.739960432053	-

b. Algorithm Request Acceptance Rate: It can be seen from Fig. 2 that the acceptance rate of the three algorithms was relatively high at the beginning, and almost any slicing request can be met with sufficient resources in the underlying network. With the rapid occupation of the underlying resources, the three algorithms have begun to experience a decline in request acceptance rates. The OLR-NS algorithm relies on reinforcement learning to accurately and reasonably allocate resources, and has the highest acceptance rate and a slow decline curve in the initial stage. In the later stage of the mapping, the request acceptance rate is in a long-term stable state, and the OLR-NS and MCRM algorithms are recombined and maintained at a relatively high state. However, as shown in Table IV, OLR-NS has a lot less time complexity than MCRM in terms of the average execution time of the algorithm. Highlight all author and affiliation lines.

c. Algorithm Resource Utilization: Fig. 3 and Fig. 4 show the performance of the three algorithms on node resource utilization and link resource utilization. The OLR-NS algorithm is superior to other algorithms whether it is at the beginning of the mapping or in a long-term stable state. It proves that the strategy network of the reinforcement learning model effectively utilizes the underlying resources during the mapping process. Through feature state extraction and training, the agent is each virtual node selects the best underlying node; it also proves that the reinforcement learning model can effectively alleviate the problem of low resource efficiency caused by time delay and reliability.

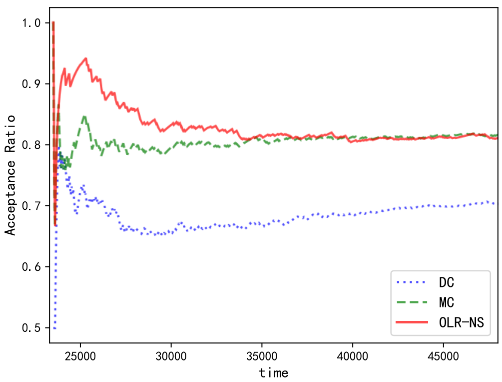

Fig. 2: Request acceptance rate.

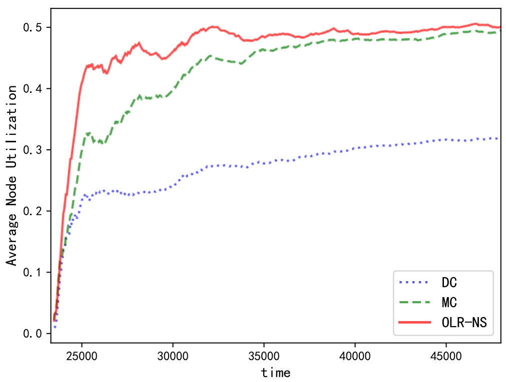

Fig. 3: Average node utilization.

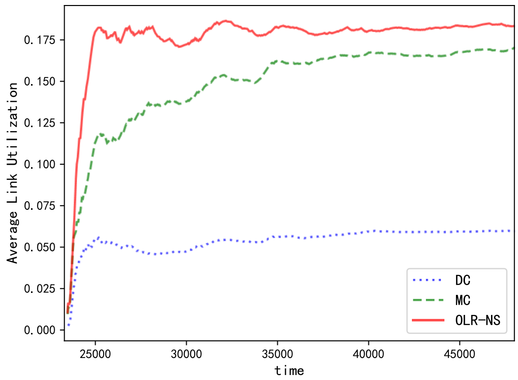

Fig. 4: Average link utilization.

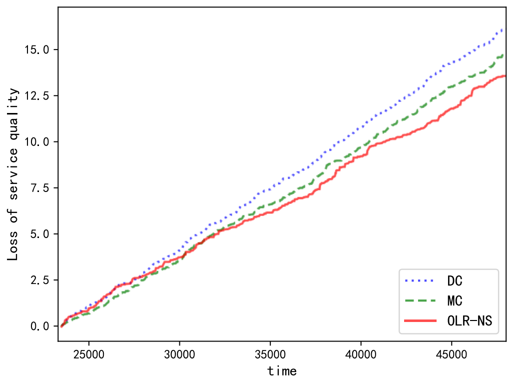

Fig. 5: Loss of service quality.

d. Algorithm Service Quality Loss: Fig. 5 shows the performance of the mapping results of the three algorithms in terms of service quality. The figure shows the difference between the delay and reliability evaluation of the mapping result and the demand of the slice request, that is, the service quality loss value. If it is less than the requested maximum tolerance value, the loss value of this mapping is 0, and an accumulative sum form is adopted at the same time, so that the result display can be more intuitive. It can be seen from the figure that the ORL-NS algorithm has a lower service quality loss. Therefore, in the actual user experience, in addition to providing a network with corresponding characteristics, it can also provide a more secure and stable network service supply. These characteristics are also in line with the URLLC application. The requirements of the scene.

e. Algorithm Revenue to Cost Ratio: Fig. 6 shows the comparison of the long-term benefit-cost ratios of the three algorithms, and the OLR-NS algorithm is in a relatively low position at all stages. This is because, in order to improve the request acceptance rate and service quality assurance of the network, many requirements for link reliability and delay have been increased in the link mapping stage, and the requirements for cost have been appropriately relaxed, resulting in a relatively low selection of mapping results. The long path increases the cost of the link, so there is no higher revenue-cost ratio between the MCRM algorithm and the DC algorithm.

5. Conclusion

This paper studies the 5G network, in order to meet the needs of different application scenarios and realize the service-oriented network slicing resource mapping problem, a network slicing mapping algorithm oriented to delay and reliability perception based on reinforcement learning is proposed. The simulation results show that compared with the DC and MCRM algorithms, the algorithm has a higher slice acceptance rate, long-term resource utilization, lower running time cost and higher quality of service guarantee. Among them, this article simply distinguishes slicing requests based on two characteristics: reliability and delay. At present, 5G has three major application scenarios, and there can be more complete and fine-grained business distinctions. Therefore, in the next step, we will continue to improve the evaluation plan for slice business types and design a better slice mapping algorithm to balance the relationship between service quality, acceptance rate, and resource utilization.

6. Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.61531013), and the key special funded projects of the national key R&D plan (No.2018YFB1800300).

7. References

[1] ITU-R M 2083-0. IMT vision-framework and overall objectives of the future development of IMT for 2020 and beyond[R]. Geneve: International Telecommunication Union, 2015.

[2] Li T, Zhu X, Liu X. An End-to-End Network Slicing Algorithm Based on Deep Q-Learning for 5G Network[J]. *IEEE Access*, 2020, PP**(99)**:1-1.

[3] Y. Wang, J. Zhang, J. Liu, et al. SCD-NS: Service Customized Deployment of Network Slicing[J]. *IEEE 6th International Conference on Computer and Communications (ICCC)*, 2020, pp. 349-356.

[4] Li Z, Lu Z, Deng S, et al. A Self-Adaptive Virtual Network Embedding Algorithm Based on Software-Defined Networks[J]. *IEEE Transactions on Network and Service Management*, 2019, vol. **16,** no. 1, pp. 362-373

[5] Guan W, Wen X, Wang L, et al. A Service-Oriented Deployment Policy of End-to-End Network Slicing Based on Complex Network Theory[J]. *IEEE Access*, 2018:19691-19701

[6] WANG R. Research on SDN-based Network Slicing Resource Mapping and Orchestration Technology[D]. Nanjing: Southeast University, 2018.

[7]] LIU G Y, SU S. Less Stringent Reliable Virtual Network Mapping Algorithm for Substrate Single Node Failure[J]. *Journal of Electronics Information Technology*, 2013, **35**(11): 2644-2649.

[8] LIU Jiajia, JIANG Zhongyuan, et al. Reliability evaluation for NFV deployment of future mobile broadband networks[J]. *IEEE Wireless Communications*, 2016, **23**(3):90–9.

[9] KONG Jian, KIM I, WANG Xi, et al. Guaranteed-availability network function virtualization with network protection and VNF replication[C]. 2017 IEEE Global Communications Conference, Singapore, 2017: 1–6.

[10] LIU X R. Research on Virtual Network Mapping Strategy Based on Survivability[D]. Beijing: Beijing University of Posts and Telecommunications, 2018.

[11] Wu D, Liu Z, Yang Z, et al. Survivability-Enhanced Virtual Network Embedding Strategy in Virtualized Wireless Sensor Networks[J]. *Sensors*, 2020, **21**(1):218.

[12] ZHAO G F, TANG L, HU Y J, et al. A Reliability-aware 5G Network Slice Reconfiguration and Embedding Algorithm[J]. *Journal of Electronics Information Technology*, 2020, (6):1478-1485.

[13] XIE B H, LIANG J, XIAO N, et al. Reliabilitybased 5G-LEO constellation network slice mapping algorithm[J/OL].*Application Research of Computers***1-5**[2021-10- 09]. https://doi.org/10.19734/j.issn.1001-3695.2021.04.0146.

[14] SU Y Z, MENG X R, ZHAO Z Y, et al. Research on Node Reliability-aware Efficient Virtual Network Embedding Algorithm[J]. *Advanced Engineering Sciences*, 2018, **50**(02):126- 132.

[15] Patounas, Georgios, Marina, et al. Network Slicing in 5G: Survey and Challenges[J]. *IEEE Communications Magazine: Articles*, News, and Events of Interest to Communications Engineers, 2017, **55**(5):94-100.

[16] Debbabi F, Jmal R, Fourati L C, et al. Algorithmics and Modeling Aspects of Network Slicing in 5G and Beyonds Network: Survey[J]. *IEEE Access*, 2020, **8**:162748-162762.

[17] H. Yao, B. Zhang, P. Zhang, et al. RDAM: A Reinforcement Learning Based Dynamic Attribute Matrix Representation for Virtual Network Embedding[J]. *IEEE Transactions on Emerging Topics in Computing*, 2021, pp.901-914.

[18] Wang C, Zheng F, Zheng G, et al. Modeling on virtual network embedding using reinforcement learning[J]. *Concurr Comput Pract Exp*. 2020.

[19] Li R, Zhao Z, Sun Q, et al. Deep Reinforcement Learning for Resource Management in Network Slicing[J]. *IEEE Access*, 2018， **4**:74429-74441.

[20] ZHAO L Y, Research on Network Slice Resource Mapping and Orchestrating Method[D]. Beijing: Beijing University of Posts and Telecommunications, 2020

[21] Nasiri A, Derakhshan F. An Agent Based Approach For Assignment Of Virtual Networks To Substrate Network For Software Defined Networking[C]// IEEE International conference on Smart Energy Grid Engineering (SEGE)2018. IEEE, 2018.

[22] Bianchi F, Presti F L. A Markov Reward based ResourceLatency Aware Heuristic for the Virtual Network Embedding Problem[J]. *Performance evaluation review*, 2017, **44**(4):57-68.