A Failover Mechanism for SFC Requests in Next-Generation Networks

Nadia Niknami, Abdalaziz Sawwan, and Jie Wu
Center for Networked Computing, Temple University, USA

***Abstract*—In 5G network technology, low latency and high data rates are expected to be achieved at an unprecedented level, enabling the development of a wide range of enhanced applications and services. Network Function Virtualization (NFV) and Multi-access Edge Computing (MEC) are crucial to meeting ambitious Quality of Service requirements. MEC reduces latency and reduces the load on transport networks by bringing computing capabilities to the edges of mobile networks, while NFV enables service providers to maximize profit via resource-efficient Quality of Service (QoS) provisioning for user-requested SFC. The challenge is to provide failover protection for Service Function Chain (SFC) at minimal end-to-end delay and with high usability. This paper presents a method for assigning NFV backup to ensure required availability and meet the deadlines for each request while maximizing utility values. Mathematical formulations are used to propose methods to optimally assign primary and backup functions to each request over a network according to associated utility functions. Simulations and testbed experiments show that the proposed failover method provides maximum utility, acceptable delay, and high availability compared to other approaches.**

***Index Terms*—Backup, Cloud Computing, Deadline-Aware Requirements, Edge Computing, 5G, Quality of Service (QoS), Service Function Chain (SFC), Virtualized Network Function (VNF).**

NOMENCLATURE

Abbreviation

MEC	Mobile Edge Computing
$MTBF$	Mean Time Between Failure
$MTTR$	Mean Time To Repair
NFV	Network Function Virtualization
QoS	Quality of Service
SDN	Software-Defined Network
SFC	Service Function Chaining
VNF	Virtual Network Function

Variables

γ_k	SFC request k
$\mathbb{C}(v_f)$	Cost of running function f on a virtual machine
$\mathbb{T}_F$	Expected time required for the request
$\mathcal{N}(\mu, \sigma)$	Normal Distribution
ϕ_k	Required deadline of request k
Ψ	Utility value
ξ	Finishing time for the request k
b_i	i-th backup virtual machine
F	Total number of service functions
f	Service functions
G_p	Physical network
G_v	Virtual network
K	Number of VMs available for each NF
M_i	i-th server
p_f^i	Reliability of virtual machine v_f^i
$R_{F-1}^{b_i}$	Optimal remaining time of execution for the rest of required request service functions after a successful execution at the node $v_{F-1}^{b_i}$
$T_{f_1,f_2}^{i_1,i_2}$	Shortest time to transfer from $v_{f_1}^{i_1}$ to $v_{f_2}^{i_2}$
T_f^i	Execution time needed by v_f^i
$U(\gamma)$	Utility function
v_f	f-th virtual function
v_f^B	VNF for v_f on Cloud
v_f^i	i-th VNF for v_f

Sets

$\mathfrak{S}_K$	Set of all possible permutations of elements $(b_1, b_2, \ldots, b_K)$
E	Set of virtual links
L	Set of physical links
M	Set of physical nodes/machines
SC	Set of ordered VNFs in a SFC
V	Set of VNFs

This research was supported in part by NSF grants CNS 2214940, CPS 2128378, CNS 2107014, CNS 2150152, CNS 1824440, and CNS 1828363.

I. INTRODUCTION

5G networks offer a wide array of services, including massive broadband, virtual/augmented reality, autonomous vehicles, real-time monitoring, and more. These services have stringent quality of service (QoS) requirements in terms of data transmission rate, latency, reliability, and mobility [1]. To address the diverse needs of 5G, network slicing has emerged

as a cost-efficient solution. This approach leverages Software-Defined Networking (SDN) and Network Function Virtualization (NFV), as well as advancements in cloud computing like Mobile Edge Computing (MEC). NFV implements network functions as Virtual Network Functions (VNFs) running on commodity hardware, while SDN separates the control plane from the data plane, enabling programmable connectivity. The combination of NFV and SDN [2] allows for network softwarization, reducing infrastructure costs, and improving flexibility, scalability, and agility. The deployment of VNFs over edge networks, as facilitated by the development of edge computing and 5G networks, maximizes their potential. Furthermore, the use of network slices on a shared physical infrastructure, as shown in Fig. 1, further enhances cost-effectiveness and allows customized services.

Each network slice consists of a MEC node and a set of VNFs, also referred to as Service Functions (SFs). These SFs, such as firewalls, network address translators, and deep packet inspections, are responsible for processing end-to-end traffic flows, meeting operator policies, and fulfilling service requirements. Service Function Chaining (SFC) defines an ordered set of SFs for specific flows, guiding the flow accordingly. Overall, the combination of NFV, SDN, and network slicing empowers operators to efficiently deliver diverse 5G services while optimizing resource utilization and service customization. In the context of SFC mapping, quality-of-service provisioning is a crucial requirement, and end-to-end delay is one of the essential metrics considered in recent mission-critical applications. There are a variety of factors that contribute to end-to-end delay, including transmission, propagation, processing, and queueing delays.

Ensuring the reliability of VNFs presents a unique set of challenges, particularly when deployed on generalized hardware as part of NFV. Unlike specialized telecommunications equipment, VNFs on such hardware carry a higher probability of faults resulting from hardware, software issues, or connectivity losses. The reliability of an end-to-end service, based on a sequence of VNFs, is intricately tied to the composition of all components [3]. Failures in any VNF within the SFC can disrupt the entire chain, causing service interruptions [4]. To address this, backup-based VNF protection, with failover mechanisms, becomes imperative. Failover, where a backup VNF seamlessly takes over the responsibilities of the primary VNF, ensures service continuity. Swift detection and reaction to failures, minimizing traffic loss, are crucial for effective backup protection, involving VNF migration and traffic redirection [5]. Local repairs are initiated by redirecting requests to the backup instance of the service function upon failure detection. However, determining the optimal number of backup VNFs for reliability remains a challenge. Redundancy is crucial for system reliability, especially in NFV-based networks where a single VNF failure can lead to a loss of service continuity [6].

Various replication mechanisms, including VNF redundancy, have been proposed to address this, but challenges persist. In the context of 5G networks, where diverse SFC

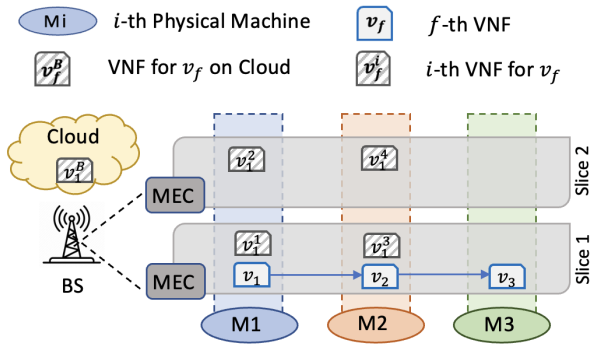


Fig. 1: Structure of 5G network with two network slices.

requests with varied requirements are common, the application of VNF redundancy alone proves insufficient. The deployment of backup VNFs in the cloud presents challenges, which could lead to increased latency and decreased performance due to added communication overhead. This study focuses on specific challenges for 5G networks and proposes a reliability guarantee failover mechanism for SFC requests. Multiple backup VNFs are strategically distributed across different network slices within 5G. In case of a VNF failure, the selection of a backup VNF considers factors such as request type and deadline. The proposed utility model, formulated as an optimization problem with reliability constraints, aims to maximize the utility value of SFC requests by considering request type, cost of backup VNFs, and other relevant factors [7]. Addressing these challenges is crucial to ensure the robustness of failover mechanisms in the dynamic environment of 5G networks.

The main contributions of this paper are as follows:

- We introduce a failover mechanism for Service Function Chains in sliced networks, designed to handle diverse use cases seamlessly. It is like a versatile tool, ensuring uninterrupted service even in challenging scenarios.
- Delving into 5G/6G networks with Multi-Access Edge Computing, we have identified potential failures and devised dedicated backup methods. Our proposed approach is designed to be resilient against both VNF failures and server failures.
- Our proposed failover mechanism intelligently selects the optimal backup VNF, minimizing downtime, and maximizing utility value.
- We provide a dynamic programming solution, turning the failover challenge into an optimization problem. Considering deadlines and reliability constraints, it is a sophisticated effective strategy for resilient failover.
- We assess the effectiveness of our suggested approaches using simulations, demonstrating that they surpass the performance of existing state-of-the-art methods.

II. RELATED WORK

It is becoming increasingly imperative for users to have ubiquitous access to 5G services, low latency, and reliable communication as 5G infrastructure is more widely deployed. There has been considerable effort put into investigating fault-

tolerant mechanisms to ensure reliable service functions in the literature [8]–[10].

A. Coordination and Optimization Strategies

Most VNF activities are highly sensitive to delays and are subject to strict delay prerequisites. Nguyen *et al.* [11] focused on co-located and geographically distributed SFC coordination by considering uncertain demand insights to preemptively factor in fluctuations in service requirements. Wang *et al.* [12] explored the optimization of end-to-end delays in mobile edge computing, introducing a user-managed online SFC orchestration framework. Chen *et al.*. [13] formulated an integer linear programming problem to minimize the total deployment cost and proposed a method for optimizing the layering strategy. Our proposed approach addresses the challenge of highly sensitive to delays by introducing a dynamic programming solution that aims to find the optimal strategy for forwarding the request and executing the SFC on it.

B. Backup Strategies and Resilience Enhancements

Li *et al*. [14] proposed a deployment and backup scheme for resource efficiency. Fan *et al*. [15] and Wang *et al*. [16] tackled SFC mapping issues, considering availability and proposing joint path-VNF backup methods. Mohan *et al*. [17] focused on embedding resilient network slices, while Wang *et al*. [18] introduced parallelized SFCs and a hybrid placement algorithm. Our proposed approach provides a failover method that is a procedure for switching from a primary network function to the best option among different possibilities for backup VNFs in the event of failure. Wu *et al*. [19] introduced a model for concurrent execution of service request VNFs, enhancing service reliability through shared backups. Qu *et al*. [20] focused on reliability-driven placement of VNF, and Karimzadeh *et al*. [21] tackled the joint placement of VNFs and backups.

C. 5G-Specific Approaches

To fulfill scale, throughput, latency, and reliability needs, 5G adopts VNFs. Zhang *et al*. [22] proposed an adaptive interference-aware approach. Wang *et al*. [23] presented a Real-Time Selection and Deployment (RTSD) algorithm to deploy SFC backups on the edge. Liu *et al*. [24] and Masoumi *et al*. [25] proposed methods to minimize communication delays and enhance network resiliency against single failures. Perez-Valero *et al.* in [26] proposed a control theory to design an auto-scaling technique for a server farm for NFV that guarantees a certain reliability while minimizing the number of active resources. They considered both the activation delay until servers become available (i.e., the wake-up or activation time) and the fallible nature of servers (which may fail with some probability) Our proposed approach considers different types of backup VNF in the case of location on slices in 5G networks. In the worst case, the assignment is to the backup function on the Cloud, which has the highest level of reliability as well as the greatest amount of delay.

Our proposed approach addresses the overlooked aspect of network function resilience in 5G. The focus on diverse backups for VNFs across network slices ensures an alternative route in the event of hardware or software failures.

III. BACKGROUND AND MOTIVATION

5G networks offer unprecedented capabilities that introduce complex challenges in ensuring reliable failover mechanisms with their advanced technologies such as MEC and Network Slicing. The inherent dynamism and heterogeneity of 5G environments, coupled with stringent QoS requirements, complicate the deployment of resilient and efficient failover strategies. These challenges are exacerbated by the necessity to balance low latency, high data rates, and the seamless provision of services across diverse applications. The integration of MEC and Network Slicing further complicates failover mechanisms due to the variable nature of edge computing resources and the need for dynamic resource allocation across slices. This section delves into the unique challenges these technologies pose to designing effective failover mechanisms in 5G networks, setting the stage for our motivation to address these critical issues.

5G networks offer significant advancements in capabilities and performance compared to previous generations. They provide low latency, high data transfer rates, and support various advanced applications such as virtual reality, autonomous vehicles, and industrial automation. Additionally, 5G networks are reliable, energy-efficient, and flexible, making them suitable for accommodating numerous connected devices and IoT applications. To meet diverse requirements, 5G employs key technologies like MEC and Network Slicing. MEC is a distributed computing paradigm that brings computing capability closer to the edge of the network, near the end user. This allows for lower latency and higher performance for applications that require real-time processing, such as augmented reality and autonomous vehicles. By moving computing resources closer to the edge, it reduces dependence on cloud infrastructure and increases the reliability of the service. MEC optimizes the performance and user experience, while Network slicing allows for efficient and flexible use of network resources [27].

The motivation of this study stems from the pressing need to address the identified challenges in failover mechanisms within the context of advanced 5G technologies. Given the critical role of failover mechanisms in maintaining service continuity and meeting QoS requirements, it is imperative to develop strategies that are not only robust and reliable but also cognizant of the 5G architecture's nuances. This includes ensuring that backup VNFs can be deployed and activated with minimal latency, leveraging the distributed computing capabilities of MEC, and effectively utilizing network slices to maintain service quality even in the event of failure. The complexity of these requirements motivates our investigation into innovative failover solutions that can meet the high standards of flexibility, reliability, and performance demanded by 5G networks.

The advanced capabilities of 5G technologies such as MEC and Network Slicing pose unique failover challenges. The distributed nature of MEC and the tailored virtual networks created by Network Slicing demand failover solutions that are both flexible and capable of quick adaptation to changing network conditions and failures. The role of VNFs especially in ensuring high availability and reliability across the 5G infrastructure becomes important. Our focus is on developing a failover mechanism that not only addresses these unique challenges but also leverages the strengths of these technologies to ensure uninterrupted service delivery.

Network Slicing enables the creation of customized virtual networks on a shared physical infrastructure, allowing efficient resource utilization and tailored service provision based on specific requirements such as low latency or high bandwidth. MEC, on the other hand, brings computing capabilities closer to the network edge, reducing latency, and enhancing performance for real-time applications like augmented reality and autonomous vehicles. By leveraging MEC and Network Slicing, 5G networks optimize resource usage and provide efficient and flexible support for various services and applications [27]. Because networks are comprised of Network Functions (NFs), and each slice of a network flexibly requires dedicated resources, the VNF technology has been identified as one of the most promising enablers for transforming 5G network slicing into a reality. NFV is the technology that enables the creation of VNFs that can be deployed and interconnected to create SFCs. SFC is a key concept in 5G networks that involves the interconnection of virtual network VNFs to create a chain of NFs that are used to provide a specific service or application.

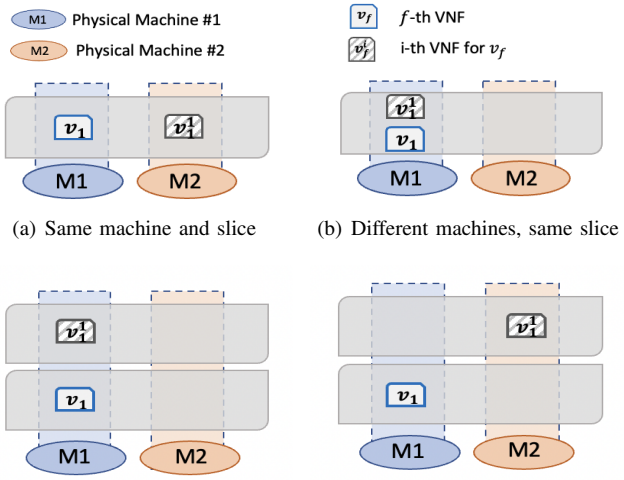
In 5G networks, SFCs are used to create virtual networks that can be customized to meet the specific requirements of different services and applications. The SFC plays an important role in the delivery of sophisticated services per slice, enabling traffic to traverse a set of ordered service functions such as firewalls, IDS, DPI, video optimizers, load-balancing servers, and NAT, among others. Since SFC is an infrastructure-independent technology, it is essential for managing and optimizing each 5G complex service based on network features and users' preferences [28]. These VNFs can be assigned to different network slices, which are created on demand to meet the specific needs of a service. When it comes to providing VNFs in a network slice, network slicing allows the creation of virtual networks with different characteristics to support different services and applications, such as low latency or high bandwidth. Furthermore, MEC technologies enable the VNFs to be deployed at the edge of the network, reducing the dependence on cloud infrastructure and increasing the reliability of the service. MEC provides low-latency, high-performance computing at the edge of the network, while NFV allows the creation of virtual networks with different characteristics to support different applications [29].

VNFs can be used to implement a wide variety of NFs, including those found in the radio access network (RAN), the core network, and the transport network. When a VNF is deployed in a 5G network, it becomes a primary VNF that is responsible for performing a specific NF. One of the key requirements for 5G is high availability at the control and data planes. There is the possibility of temporary unavailability of NFs due to misconfiguration or software and hardware failures.Availability problems in NFVs can be divided into hardware failure (processor, memory, storage, and network interface) and software failure (host operating systems, hypervisor, virtual machines, and VNF software configuration). In addition to the primary VNF, it's also possible to deploy one or more backup VNFs. A backup VNF is a secondary VNF that can take over the function of the primary VNF in the event of failure or maintenance. By utilizing redundancy, high levels of reliability can be achieved. In the event of failure of the primary VNF, the backup VNF can automatically be activated and take over the responsibility of the primary VNF and continue the network operation. This process is called *failover*, which is the process of switching over to a backup VNF when the primary VNF fails.

For instance, in the RAN network slice, a primary VNF could be a base station controller (BSC) that manages the communication between a device and a base station, and a backup VNF could be a virtualized BSC that can take over the function of the primary BSC in the event of failure. Similarly, in the core network, the primary VNF could be the Mobility Management Entity (MME) which is responsible for handling mobility management of the devices in the network, and the backup VNF could be another virtualized MME that can take over the function of primary MME in the event of failure. The use of VNFs and backup VNFs provides a high degree of flexibility and scalability in 5G networks. It makes it possible to easily deploy new NFs, as well as easily upgrade or replace existing NFs, making the network more reliable and resilient. Therefore, an efficient failover mechanism is essential in 5G networks due to the wide variety of service requests with varying requirements. We propose a method for assigning backups to service requests in order to meet the request in a minimum of time and with the highest utility.

IV. The Proposed Failover Mechanism

In this section, we present our failover approach to ensure the required request availability while maximizing utility. A failover method is a procedure to switch from a primary network function to a backup function in the event of failure or maintenance. The 5G core network is designed using NFV and SDN to construct network slices that meet a variety of needs. A virtualized environment consists of a series of virtual network functions that are combined in an orderly manner. Failure of a VNF will result in an interruption of service, thus a backup VNF is required to ensure the reliability of slices. Some VNFs are dedicated to slices, while others are shared across multiple slices. Each type of VNF instance has a specific availability, and SFCs require reliability. VNFs can be deployed in either a central cloud for scalability or an edge cloud for reduced latencies. Network functions can be distributed across different slices to optimize physical network resource usage. NFV enables the abstraction of physical infrastructures



(a) Same machine and slice (b) Different machines, same slice

(c) Same machine, different slices (d) Different machines and slices

Fig. 2: Possible backup locations.

into a logical virtual network, facilitating the distribution of network resources and functions among network slices. However, VNFs can experience temporary unavailability due to misconfiguration or hardware and software malfunctions. To address this, an active-standby deployment can be used, where backup instances are activated in case of failure.

Fig. 2 illustrates different possibilities for backup VNFs. In Fig. 2(a), the primary and backup VNFs are located on the same machine and on the same slice. This backup method provides the least amount of delay. Fig. 2(b) illustrates the case where the primary and backup VNFs are assigned to different machines, but they are part of the same slice. In the event of hardware failure, the backup VNF would have a higher level of reliability. Fig. 2(c) shows that the backup VNF is being assimilated on the same machine but in a different slice. The backup VNF provides higher reliability but greater delay if the primary VNF fails due to software failure. Assigning the primary and backup VNF to different machines and slices is illustrated in Fig. 2(d). The highest level of reliability as well as the highest degree of delay can be found in the type of assignment to cloud backup.

A. Problem Formulation

The physical network is a building block to host multiple service functions and monitoring functions on a single physical infrastructure, which is represented as an undirected graph $G_p = (M, L)$, where M denotes the set of physical nodes/machines, and L denotes the set of physical links which interconnect the physical nodes. There is a limit to the amount of resources available on each physical node, and to the bandwidth available on each physical link. The virtualization of physical network resources enables the provision of multiple services and monitoring functions from a single physical infrastructure node. In NFV infrastructure, we model the virtual network as an undirected graph $G_v = (V, E)$. The set of VNFs V includes the various virtualized network functions (e.g., load balancer, firewall, proxy, and network functions) instantiated on virtual machines, and the set of virtual links E represents the connections between these functions. Each virtual network function $v \in V$ requires a certain amount of resources C_v to process the traffic according to the type of service, and each virtual link $(v_f, v_{f+1}) \in E$ connects pairs of VNFs v_f and v_{f+1}.

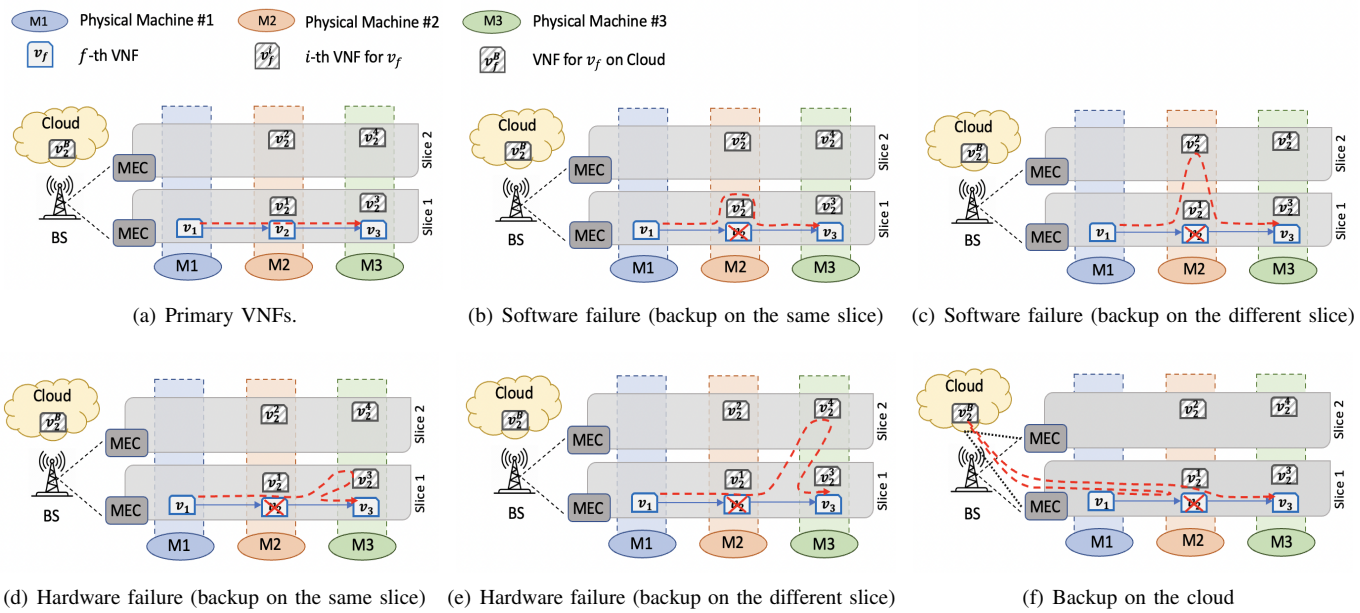
This virtual network is part of the overall cloud network infrastructure, which encompasses both physical and virtual network resources. In a 5G network, network functions are provided in network slices. For each network function, one of the provided functions is a primary function and others are backup functions. We can consider a network of virtual machines, where each one of them is capable of providing F sets for a certain network function. A request arrives on the network and requires a certain order of the service function or an SFC to be executed.

Definition 1. (SFC Request) Each SFC request is denoted as γ_k, which consists of multiple VNFs v_f connected in a specific order. A request can be defined as $\gamma_k = (v_f \rightarrow v_{f+1} \rightarrow \cdots \rightarrow v_F)$. If all the v_f in the requested SFC were executed successfully, γ_k can be considered as a satisfied request. There is a deadline ϕ_k for each request γ_k. The deadline is an indicator of when a request is supposed to be completed. It is important to answer a request in a minimum amount of time.

For example, the request may require the service function chain $(v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4)$, which means it needs to be served by a virtual machine from the ones that can execute service function 1, then one that can execute service function 2, and so on.

Every virtual network function v_f is connected to other v_f^i through B edges, to all the virtual network function v_{f-1}^i in the previous set of virtual network function (if there is such set) through B edges, and to all the virtual network function v_{f+1}^i in the next set of virtual network function through B edges (if there is such set). The transmission time between two v_f^i and v_f^j is associated with a specific time $T_f^{i,j}$. Every virtual machine v_f^i has an associated reliability or success probability p_f^i, which means that with this probability, the virtual machine succeeds in executing the virtual network function v_f. The request must be executed successfully by at least one virtual network function in the first set of virtual network functions and then must be executed by at least one virtual network function in the second set of virtual machines, and so on until it reaches the final set of virtual network functions in the required order of service function execution.

It is possible to decompose an SFC into its components, and the reliability can be determined using a network management system, such as SDN. Reliability is determined on the basis of the amount of time a component has been available compared to the total amount of time the network has been operational. The mean time between failures (MTBF) and mean time to repair (MTTR) are commonly used to measure uptime and downtime [30]. Therefore, the reliability of a VNF can be expressed as $p = Uptime/(Uptime+Downtime)$ or, in other



(a) Primary VNFs. (b) Software failure (backup on the same slice) (c) Software failure (backup on the different slice)

(d) Hardware failure (backup on the same slice) (e) Hardware failure (backup on the different slice) (f) Backup on the cloud

Fig. 3: Possible backup assignment for $\gamma = (v_1 \rightarrow v_2 \rightarrow v_3)$.

words, $MTBF/(MTBF + MTTR)$.

Definition 2. (Reliability of SFC.) Each SFC consists of a number of VNFs, and ideally each component should operate properly. Every VNF failure is independent of the others.A serial reliability can be described by $p = \prod_{v_f^i} p_f^i$, where v_f^i indicates the i-th VNF in a SCF. Therefore, any VNF failure will result in the failure of the service as a whole and an interruption of the network slice service. If the primary VNF fails, the backup VNF can be replaced immediately. As a backup VNF, its reliability cannot be lower than the reliability of a primary VNF, which provides a greater level of backup efficiency overall.

To illustrate the problem, we will use an example here. Suppose that there is a request for $\gamma = (v_1 \rightarrow v_2 \rightarrow v_3)$. This request has a deadline of $\phi = 100s$, and the utility value would be zero if the deadline was missed. If there is a failure for each of the VNFs in the requested service functions, there are some backup options. We are supposed to see a failure for v_2. Fig. 3 shows five different backup methods to handle network components failure in a 5G network with two network slices. It shows the backup method when the primary and backup are in the same/different machines and in the same/different slices. Fig. 3(a) displays the requested SFC γ fulfilled by the primary version of the requested VNFs. In the case of failure in v_2, as shown in Fig. 3(b), backup to the primary of VNF v_2, which is v_1^2, is assigned in the same physical machine $M2$ and in the same $Slice1$. This backup method provides the lowest delay. In the case of unavailability for v_1^2, as shown in Fig. 3 (c), the backup to the primary v_2 is assigned in the physical machine but a different network slice, which is $Slice2$. In the case of hardware failure on a physical machine $M2$, as shown in Fig. 3(d), backup to the primary v_2 is assigned in a different physical machine $M3$ but to the same network slice $Slice1$. This method is acceptable for reliability-aware requests because the primary and backup are in different physical machines. Fig. 3(e) shows assigning backup of v_2 to the given request on a different physical machine $M3$ as well as a different network $Slice2$.

If backup cannot be achieved across machines and various VNFs within slices, an alternative backup will be supplied in the cloud. The reliability in the cloud is 100%. Fig. 3(f) shows the scenario where the backup for v_2 is provided in the cloud. There is no doubt that this is the worst-case scenario in terms of delay, since the delay between edge and cloud is larger than the delay between edge servers, especially when a network is based on 5G technology. Backup method 1 and method 2 can handle only VNF failure, whereas backup method 3 and method 4 can handle VNF failure and hardware failure. We need to take into account the reliability of each backup as well as the time spent on the transmission and execution of functions. The one that provides higher utility for responding to the given request would be selected as the backup for v_2 in SFC.

V. Solution of the Problem

In this section, we present the dynamic programming solution that aims to find the optimal strategy for forwarding the request and executing the SFC on it. We start by abstracting the problem by mapping it to a fully connected graph where each node is denoted by v_f^i, which is the i-th virtual machine available for service function f. The number of services in the SFC required by the request is F. The total number of nodes for each service function is K. Without loss of generality, we consider the cloud to be one of the virtual machines that has high values of execution time and transfer time to and from all the virtual machines. We can consider the problem as a Markov decision process (MDP). Each state of the MDP

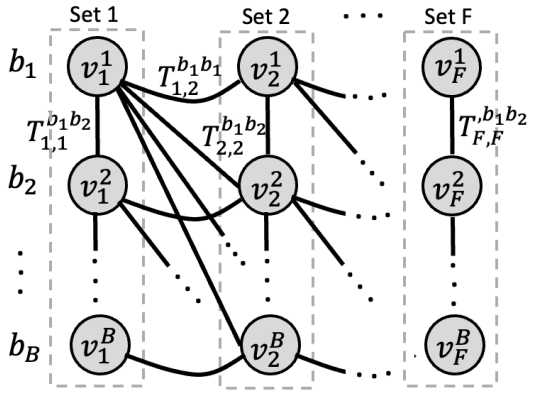


Fig. 4: All the possible back up VNF.

represents a subset of the service function chain that has been executed, and each action represents a choice of the virtual machine to execute the next service function in the chain. The goal of this MDP is to find the optimal policy in which the expected reliability times transmission time of the service function chain is minimized.

Fig. 4 shows MDP for the given problem. For each service function required by the request, the primary node for executing it has the index value of b_1, and the order of the backup nodes to execute the service function in case of failure is $b_2 \rightarrow b_3 \rightarrow b_K$, where b_i is the index of the node with the i-th order to execute the service function in case of failure. b_K is typically set to the cloud due to the inefficiency of time in using it. We refer to node b_0 as the node at which the request resides initially within the service function nodes. This node does not necessarily attempt to execute the service function first among the other nodes for the same service function.

In other words, we have K nodes that are capable of executing a service function f, referred to as $v_f^1, v_f^2, \ldots, v_f^K$. At the beginning, the request resides at one of those nodes with index b_0, and then the request is transferred to the node with index b_1 to attempt executing the service function f, and in case of failure, the request is transferred to b_2 for execution, and so on until the request reaches b_K if all previous nodes fail, which is typically the cloud node which is considered to be 100% reliable in our model. This means that the expected total time for the request to have a service function f successfully executed is the time required to transfer the request from the node $v_f^{b_0}$ to $v_f^{b_1}$, and then the expected time to execute the request successfully from that node is the probability of success times the execution time of the node $v_f^{b_1}$ plus the probability of failure times the expected time in case of failure.

Based on this idea, we can start constructing the dynamic programming recursion to evaluate the optimal total expected time to execute a request. Starting from the last service function required by the request, indexed F, and considering that the request initially resides at the node $v_F^{b_0}$, and considering a specific order of execution $b_1, b_2, \ldots, b_K$, the expected time required for the request to be fulfilled is shown in the nested

Algorithm 1 Optimal Path and Order of Backups Solution through Dynamic Programming

Input: $G, \gamma, \{T_{f_1,f_2}^{i_1,i_2} | \forall (v_{f_1}^{i_1}, v_{f_2}^{i_2})\}, \{p_f^i | \forall v_f^i\}$
Output: Optimal path for the lowest expected time
Initialization: $R_F^x = 0 \quad \forall x \in \{0, 1, \ldots, B\}$
Get the order of the service function chain from γ and reindex it as $1, 2, \ldots, F$
for f in $F, F-1, \ldots, 1$
  for b_0 in $1, 2, \ldots, B$
    if $f \neq F$ **then**
      $R_f^{b_0} \leftarrow \min_{b \in \{0,\ldots,K\}} (T_{f,f+1}^{b_0,b} + \mathbb{T}_{f+1}(b))$
    for $(b_1, b_2, \ldots, b_K)$ in $\mathfrak{S}_K$
      Calculate $\mathbb{T}_f(b_0, b_1, b_2, \ldots, b_K)$ from Eq. (5)
    $\mathbb{T}_f(b_0) \leftarrow \min_{(b_1,\ldots,b_K) \in \mathfrak{S}_K} \mathbb{T}_f(b_0, b_1, b_2, \ldots, b_K)$
    $\mathsf{OPT}_{v_f^{b_0}} \leftarrow \arg\min_{(b_1,\ldots,b_K) \in \mathfrak{S}_K} \mathbb{T}_f(b_0, b_1, b_2, \ldots, b_K)$
return $\min_{b_0 \in \{1,\ldots,B\}} \mathbb{T}_1(b_0)$ and corresponding $\{\mathsf{OPT}_{v_f^{b_0}} | \forall f\}$.

formula as:

$$\begin{aligned}\mathbb{T}_F(b_0, b_1, b_2, \ldots, b_K) &= T_{F,F}^{b_0,b_1} + p_F^{b_1} \cdot T_F^{b_1} + (1 - p_F^{b_1}) \\ &\cdot (T_F^{b_1} + T_{F,F}^{b_1,b_2} + p_F^{b_2} \cdot T_F^{b_2} + (1 - p_F^{b_2}) \cdot (\ldots)) \\ &= \sum\nolimits_{i=1}^{K} (\prod\nolimits_{j=1}^{i-1} (1 - p_F^{b_j}) \cdot (T_F^{b_{i-1}} + T_{F,F}^{b_{i-1},b_i} \\ &+ p_f^{b_i} \cdot T_F^{b_i})).\end{aligned} \tag{1}$$

Now, we refer to the lowest expected time for execution when the request is at node $v_F^{b_0}$ by $\mathbb{T}_F(b_0)$. The value of this lowest expected time is by definition formulated as:

$$\mathbb{T}_F(b_0) = \min\nolimits_{(b_1,\ldots,b_K) \in \mathfrak{S}_K} \mathbb{T}_F(b_0, b_1, b_2, \ldots, b_K), \tag{2}$$

where $\mathfrak{S}_K$ refers to the set of all possible permutations of elements $(b_1, b_2, \ldots, b_K)$. We refer to the best permutation (order of execution) of those elements given that the request resides initially at node $v_F^{b_0}$ by $\mathsf{OPT}_{v_F^{b_0}}$. Now, considering the second last service function (i.e. service function with index $(F-1)$, in order to formulate the minimum expected time until the end, given that the request initially resides at $v_{F-1}^{b_0}$ and with an assigned order $b_1, b_2, \ldots, b_K$ of nodes' indexes to execute the service function, this time will be formulated as:

$$\begin{aligned}\mathbb{T}_{F-1}(b_0, b_1, b_2, \ldots, b_K) &= \sum\nolimits_{i=1}^{K} (\prod\nolimits_{j=1}^{i-1} (1 - p_{F-1}^{j}) \\ &\cdot (T_{F-1}^{b_{i-1}} + T_{F-1,F-1}^{b_{i-1},b_i} + p_{F-1}^{b_i} \cdot (T_{F-1}^{b_i} + R_{F-1}^{b_i}))),\end{aligned} \tag{3}$$

where $R_{F-1}^{b_i}$ refers to the optimal remaining time of execution for the rest of required request service functions after a successful execution at the node $v_{F-1}^{b_i}$, which is, by definition, formulated as shown as follows:

$$R_{F-1}^{b_i} = \min_{b \in \{1,\ldots,K\}} (T_{F-1,F}^{b_i,b} + \mathbb{T}_F(b)), \tag{4}$$

which is the minimum of the possible remaining times that equal the summation of the time needed to transfer the request

to a specific node in the next service function and the minimum time for executing the request in the next service function, since the request would initially reside at that node. Now, we have everything set up to evaluate the minimum expected time to execute all the service functions required by the request that are indexed $1, 2, \ldots, F$. The minimum expected time would be the least value of $\mathbb{T}_1(b_0)$ among all possible values of b_0, which corresponds to the initial location of the request in the first service function $v_1^{b_0}$. Eqs. (5)-(7) give the general formulations of the optimal expected times.

$$\mathbb{T}_f(b_0, b_1, b_2, \ldots, b_K) = \sum_{i=1}^{K}(\prod_{j=1}^{i-1}(1 - p_f^{b_j}) \cdot (T_f^{b_{i-1}} + T_{f,f}^{b_{i-1},b_i} + p_f^{b_i} \cdot (T_f^{b_i} + R_f^{b_i}))), \tag{5}$$

$$R_f^x = \min_{b \in \{1,\ldots,K\}} (T_{f,f+1}^{x,b} + \mathbb{T}_{f+1}(b)), \tag{6}$$

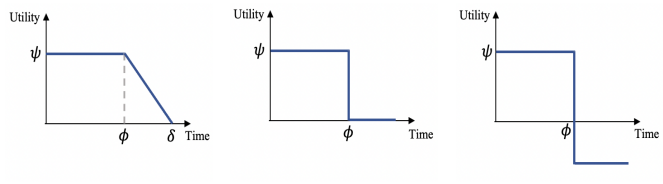
$$\mathbb{T}_f(b_0) = \min_{(b_1,\ldots,b_K) \in \mathfrak{S}_K} \mathbb{T}_f(b_0, b_1, b_2, \ldots, b_K). \tag{7}$$

A. Algorithm Overview

In this subsection, we demonstrate our algorithm that solves the problem, analyze it, and prove its correctness. To this end, we propose the dynamic programming algorithm that builds the solution block by block by evaluating the optimal expected remaining time starting from the last service function and using that solution to evaluate the optimal expected remaining time for the service function before it, and so on until we reach the optimal expected remaining time starting from the first service function required in the service function chain of request γ. Algorithm 1 exhibits our optimal solution for the problem, and this solution is devised using a variation of the dynamic programming method that employs probability analysis. Theorem 1 shows the correctness of the algorithm.

Theorem 1. The solution shown in Algorithm 1 is the optimal solution that produces the minimum expected time for a request γ to be executed and the optimal order of backups.

Proof. The algorithm starts by taking the graph (network) G as input, alongside the request γ comprised of its requested service function chain, and the set of the probability of success for every node $\{p_f^i | \forall v_f^i\}$, and the set of the shortest transfer time between every pair of nodes $\{T_{f_1,f_2}^{i_1,i_2} | \forall (v_{f_1}^{i_1}, v_{f_2}^{i_2})\}$. This set of shortest transfer times between the nodes is typically given. If it is not given, then it needs to be evaluated. That can be done using all-pair the shortest path algorithms like Floyd–Warshall algorithm [31]. Algorithm 1 outputs the optimal path comprised of the order of execution for each service function and the minimum corresponding time. That is done by solving the recursion shown in Eqs. (5)-(7). The initialization is done by setting the optimal remaining time of execution after executing the last service function to zero, and with the consideration of the correct order of the service function chain that is required by the request γ.



(a) Soft deadline (b) Firm deadline (c) Hard deadline

Fig. 5: Utility functions based on the sensitivity to deadline.

Algorithm 1 aims to build the set of values $\{\mathbb{T}_1(b_0) | \forall b_0\}$ which requires the set of all the $\{\mathbb{T}_f(b_0) | \forall b_0, f\}$. The algorithm iterates over the service functions from the last one to the first one, evaluating the optimal expected remaining time for all initial states at each service function. The build-up of those values incrementally adds from the last network slice to the first one. Iterating over all service functions, iterating over all the nodes for each service function as a possible initial node for the request to reside at, and iterating over all the possible permutations for the nodes $\mathfrak{S}_K$ that is capable of executing the service function for each initial node at each SFC are sufficient to test all possible ways from each service function nodes as the beginning point efficiently. Thus, the Algorithm 1 produces an optimal solution as it builds the best path by exhausting all the possible partial paths of how to execute the request most efficiently. □

B. Complexity Analysis

Regarding the time complexity of the solution, we can observe that line 1 of the algorithm iterates F times, and line 2 iterates B times which, as assumed in our model considered constant, as the number of backups for each SFC is typically very limited and consists of very few possible backups before the cloud. This is the reason why iterating over the set $\mathfrak{S}_K$ is not considered computationally significant, as the value of B is considered not to grow with the input. Evaluating Eq. (5) needs a time of $O(B^2)$ given that the needed values of R_f^x are handy. Evaluating Eq. (6) needs a time of $O(K)$ given that the needed values of $\mathbb{T}_f(b)$ are handy. Evaluating Eq. (7) needs a time of $O(B)$ given that the needed values of $\mathbb{T}_f(b_0, b_1, \ldots, b_K)$ are handy. This brings the total complexity of the solution to $O(F \cdot B^2 \cdot |\mathfrak{S}_K|)$ given the use of the proper data structures that enable the instant lookup of the values evaluated beforehand.

C. Extension Problem

As an extension, we can consider different types of requests. Each type of request has a different associated utility function. The utility value for finishing time larger than the deadline ($\xi_k > \phi_k$) depends on the type of request. Such an SFC request can be defined as $\gamma_k = (SC, \phi_k, U)$, where SC is the set of ordered VNFs in a SFC, ϕ_k is the deadline for responding to the given request γ_k, and U is the utility function for the request γ_k.

The utility function for each request type could be introduced as a function of the difference between the com-

pletion time and the deadline, and the objective function could be defined to maximize the overall utility value of the system. Various scenarios highlight the importance of understanding different types of deadlines, ranging from soft deadlines with manageable consequences to hard deadlines with critical implications. Let's delve into some real examples to elucidate these distinctions: The 5G network optimizes video streaming for an upcoming event with a *soft deadline* for algorithm implementation. Delayed optimization may briefly lower video quality, but with no critical consequences. The utility function gradually decreases with delays, making a soft deadline miss less critical, yet impacting request utility. A *hard deadline* mandates reducing communication latency for safe autonomous vehicle operations. Missing it may cause accidents and disruptions, leading to dire consequences or penalties. Emergency services rely on the 5G network with a *firm deadline* for a system upgrade. Missing the deadline could have severe consequences, potentially resulting in a complete loss of utility for emergency coordination and response. Fig. 5 shows the different types of deadlines for requests in the case of missing deadlines. We can define a utility function for the soft real-time request γ_k as $U(\xi_k, \phi_k) = \Psi$, where $\xi_k \in [0, \phi_k]$, and it equals to $(1 - (\xi_k - \phi_k)/(\delta_i - \phi_k))\Psi$ for $\xi_k \in (\phi_k, \delta_i]$. The objective function includes the utility function and the cost of deploying functions is as follows:

$$\max_{(b_1,\ldots,b_K)\in \mathfrak{S}_K} \quad \mathbb{U}(\gamma_k) - \sum\nolimits_{v_f \in SC_i} \mathbb{C}(v_f), \tag{8}$$

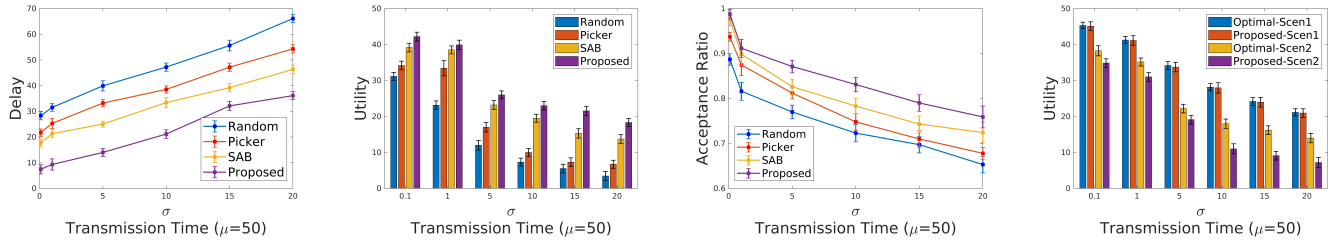
where $\mathbb{U}(\gamma_k)$ is the expected value function that expresses the value that would be granted to the system upon completion of its associated request as a function of the completion time. The value of $\mathbb{C}(v_f)$ represents the cost of running function f on a virtual machine, and it includes the deploying physical server and the cost of installing the virtual function. The utility function may not be the same for different requests with different delay sensitivities. The value for the utility function depends on the type of request γ_k, deadline ϕ_k, and how long it takes to respond to this request. The expected utility value depends on the reliability probability of sequence VNFs. By considering the utility function as a function of time for SFC requests and cost for VNFs, the problem would be an NP-hard problem.

In the failover mechanism proposed in our paper, link failures are treated as infinite delays in the affected links. This means that when a link failure occurs, the algorithm recalculates the path and order of backups to ensure that the deadline-aware SFC requests can still be satisfied. By treating link failures as infinite delays, the mechanism ensures that the backup paths selected are not affected by the failed links and are still able to meet the deadlines of the SFC requests. This approach improves the reliability of the network by providing a failover mechanism that can quickly and effectively respond to link failures without compromising the timeliness and quality of service of the SFC requests. Overall, the proposed failover mechanism provides a robust solution for ensuring the delivery of deadline-aware SFC requests in the presence of link failures.

VI. Evaluation

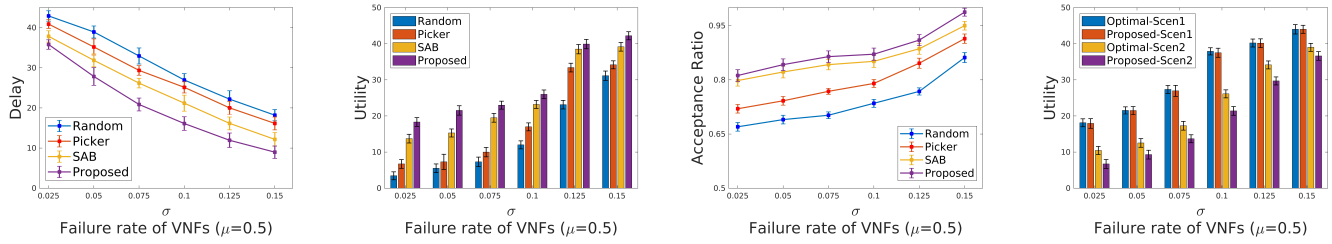
We assess the performance of our proposed backup method for the 5G network. Our network consists of four network slices, namely a control network and a data network, which operate as the SDN components in our simulation. Each server node has the capability to host specific types of VNFs v_f. We assume that the servers in the edge network are interconnected, and all servers are connected to the cloud. When handling SFC requests, the source and destination server nodes are randomly chosen. SFCs are composed of a random chain of 1 to 4 VNFs from a set of five types [32]. If no suitable VNFs are available in the edge networks, the SFC is assigned to the cloud, where the VNF reliability is considered to be 100%. We have K nodes that are capable of executing a service function f, referred to as $v_f^1, v_f^2, \ldots, v_f^B$. We assumed K is 10 in our experiment. The Python 2.7 implementation of the algorithms takes place on a laptop equipped with an Intel Core i7-8550U processor, operating at 1.80GHz (with a turbo frequency of 1.99GHz), and 24 GB of RAM. For generating the 5G network topology, the GT-ITM tool is used. In each 5G network slice, the edge and core cloud servers are categorized based on their distance from end-users. Edge cloud servers are configured with CPU resources in the range of 10 to 40 units and memory resources of 1 to 16 units. Core cloud servers have CPU resources ranging from 20 to 200 units and memory resources between 16 and 64 units. The CPU and memory resource requirements for each VNF follow random distributions within the intervals of $[1, 20]$ units and $[1, 4]$ units, respectively, as specified in [22]. To evaluate the robustness of our approach across different input parameters, we conducted experiments under two scenarios with varying cost distributions. We then assessed the effectiveness of our approach by examining transmission times, the number of SFC requests, and VNF failure rates within these scenarios. By systematically altering these input parameters, we performed a sensitivity analysis to gauge the impact of these changes on our approach's outcomes. Our experimental findings shed light on the trade-offs between transmission time, the number of SFC requests, and VNF failure rates, and how these trade-offs differ under different scenarios. The number of incoming SFC requests are ranging from 100 to 500. The reliability of each primary and backup VNFs is under normal distribution $p \sim \mathcal{N}(\mu = 0.5, \sigma)$.

Suppose that all physical nodes have the same level of reliability, which is 0.9. Since routing failure is none of our concern, we assumed link availability is 100%. The processing time of a request for each VNF is $\sim \mathcal{N}(\mu = 50, \sigma)ms$, and the transmission delay of each hop in the same slice is set to $1ms$. Each VNF may have a maximum of four backups. Overall, our sensitivity analysis highlights the importance of considering the variability of input parameters and the potential impact of different cost scenarios on the performance of our approach. By conducting such an analysis, we are better equipped to optimize our approach and make informed decisions based on the specific requirements and constraints of each scenario.



(a) (b) (c) (d)

Fig. 6: Evaluation under different amounts of time for 200 SFC requests with a monotonic utility function. In parts(a)-(c), the costs are constant. In part(d), *scenario 1* is cost $\sim \mathcal{N}(\mu = 10, \sigma = 1)$ and *scenario 2* is cost $\sim \mathcal{N}(\mu = 100, \sigma = 20)$.



(a) (b) (c) (d)

Fig. 7: Evaluation under different failure rates of VNFs for 200 SFC requests with a monotonic utility function. In parts(a)-(c), the costs are constant. In part(d), *scenario 1* is cost $\sim \mathcal{N}(\mu = 10, \sigma = 1)$ and *scenario 2* is cost $\sim \mathcal{N}(\mu = 100, \sigma = 20)$.

To eliminate random factors, the simulation is performed 100 times, and the average values are used to represent the results. We compare our proposed method with three algorithms: the *Picker*, which is based on the joint backup strategy where two VNFs with the lowest reliability are selected for joint backup each time until the reliability requirement is met [33] , the *SAB*, which is a self-adapting scheme that efficiently static backup instances and dynamic ones over both the edge and the cloud [34], and the *Random* algorithm, which randomly deploys the backup instances to the request. We evaluate the performance of the proposed model in cases where the resources of nodes are limited for numerous requests so that not all requests can be accepted. Each link has enough bandwidth to hold all requests. To evaluate the approach in the case of having different types of requests with utility and cost, we consider different kinds of costs for VNFs. We suppose the different range of cost.

A. Impact of Transmission Time

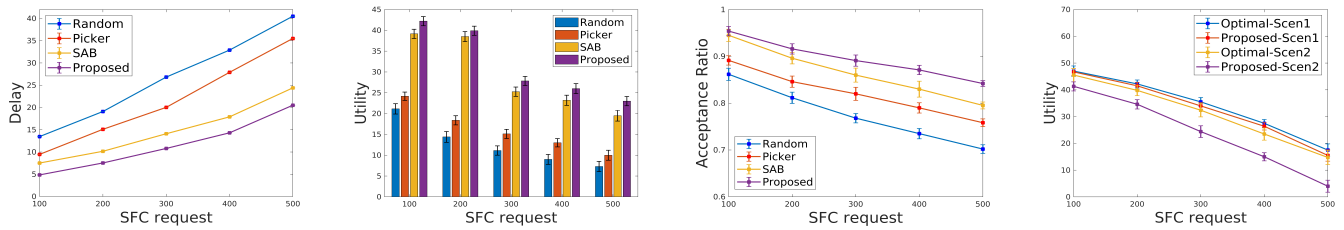
Fig. 6 shows the impact of transmission and process time on delay, utility, and average acceptance ratio. Fig. 6(a) illustrates even in the case of large transmission time, the proposed approach can find the best assignment of VNFs with a minimum amount of delay for the given request. Fig. 6(b) shows the evaluation of methods for utility value. Although under large transmission time, the utility value would reduce, the proposed method has the best utility value among *Picker*, *SAB*, and *Random*. Fig. 6(c) presents the evaluation in the case of acceptance ratio when there is varying transmission and process time. Compared to other methods, the proposed method has a higher acceptance ratio even when there is a high transmission time for VNFs. Fig. 6(d) shows the performance of proposed methods compare to the optimal under two scenarios. When the amount of costs are close to each other, the proposed method and *SAB* perform similarly in terms of utility value. However, the proposed method is not effective when the costs are in the range of *scenario 2*.

B. Impact of Failure Rate of VNFs

Fig. 7 shows the impact of the failure rate of VNFs on delay, utility, and average acceptance ratio. Fig. 7(a) illustrates the proposed approach appears capable of addressing large failure rates for VNFs more effectively than other methods. Fig. 7(b) shows there is a decrease in utility as a result of the higher failure rates for VNFs. Using the proposed method, it is possible to respond to the SFC request in a manner that maximizes the utility of the request. Fig. 7(c) illustrates the acceptance ratio when there are different amounts of failure rates for VNFs. For a low rate of failures, the acceptance ratio is close to 1 for the proposed method. When there is a higher failure rate, the acceptance ratio for all methods would be reduced. Although *SAB* and the proposed method produce similar results, the proposed approach is more effective. Fig. 7(d) compare the output of the proposed method with the optimal output when there are different failure rates and different amount of costs for VNFs. If there are no high failure rates for VNFs under *scenario 1*, the results of the proposed method are optimal. Similar to our previous results, the proposed method is unable to achieve optimal results under *scenario 2*.

C. Impact of Number of SFC Request

Fig. 8 shows the impact of the number of SFC requests on delay, utility, and acceptance ratio. The number of successfully provisioned SFC requests that satisfied the end-to-end latency and reliability requirements to the total number



(a) (b) (c) (d)

Fig. 8: Evaluation on the different number of SFC requests when reliability for VNFs is $\sim \mathcal{N}(\mu = 0.5, \sigma = 0.1)$. In parts(a)-(c), the costs are constant. In part(d), *scenario 1* is cost $\sim \mathcal{N}(\mu = 10, \sigma = 1)$ and *scenario 2* is cost $\sim \mathcal{N}(\mu = 100, \sigma = 20)$.

of SFC requests is called *Acceptance Ratio*. Comparing the proposed approach to the other methods, Fig. 8(a) displays the delay for different methods when there is a varying number of SFC requests. As a result of the proposed method, the varying number of SFC requests can be handled with the smallest delay value. There is the highest delay associated with the *Random* method. Fig. 8(b) illustrates the evaluation of methods for utility value. Even with many SFC requests, the proposed approach exhibits better performance.

To understand how the *Picker* and *SAB* work, we evaluate the Acceptance Ratio for methods. The number of successfully provisioned SFC requests that satisfied the end-to-end latency and reliability requirements to the total numbers of SFC requests is called *Acceptance Ratio*. Since a request can be accepted if and only if the requirements of given requests can be met, the rationale behind this experiment is that an algorithm with better resource efficiency can accept more requests. Fig. 8(c) illustrates the acceptance ratio for SFC requests in the case of different numbers of SFC requests. For a few requests, it is close to one for the proposed method and *SAB*. For many requests, the acceptance ratio is reduced to 0.86 and 0.79 for these two methods respectively. Accordingly, we can conclude that *SAB* provides similar performance to the proposed method when handling small numbers of requests. Fig. 8(d) shows the average utility value for the optimal and proposed method under *scenarios 1* and 2. With *scenario 1*, we can conclude that the outputs of the proposed strategy are very close to the optimal outcome. However, with the *scenario 2* the proposed method cannot handle large numbers of requests effectively.

VII. Conclusion

In 5G networks, with their low latency and versatility, technologies like NFV enable providers to deploy services suitable for diverse uses. However, VNFs' reliability is lower than traditional hardware due to software and hardware risks, addressed by redundant backup solutions. Ensuring dependable service provision is critical in 5G-enabled NFV networks due to the impact of SFC component failure on services and revenue. Primary VNF placement alone does not guarantee service continuity. This paper has addressed the challenge of providing reliable network services with minimized spending time. We have introduced a backup redundancy model and formulated the reliability-aware service chaining problem. By considering various transmission and processing times, we have developed different backup strategies to enhance the reliability of SFCs. The proposed approach takes into account VNF reliability, assignment delays, and the sensitivity of meeting deadlines for different request types. Through extensive experimentation under diverse scenarios, our method has demonstrated its effectiveness in terms of delay, utility, and SFC availability.

References

[1] J. Yao, Y. Yu, X. Chen, and P. Zhou, "5g smart grid network slice backup method based on the quality of service," in *Proc. of 6th IEEE Conference on Information Technology and Mechatronics Engineering (ITOEC)*, vol. 6, 2022, pp. 1057–1061.

[2] N. Niknami and J. Wu, "Enhancing load balancing by intrusion detection system chain on sdn data plane," in *Proc. of IEEE Conference on Communications and Network Security (CNS)*, 2022, pp. 264–272.

[3] L. Qu, C. Assi, K. Shaban, and M. Khabbaz, "Reliability-aware service provisioning in nfv-enabled enterprise datacenter networks," in *Proc. of 12th IEEE International Conference on Network and Service Management (CNSM)*, 2016, pp. 153–159.

[4] D. M. Manias, A. Chouman, J. Naoum-Sawaya, and A. Shami, "Resilient and robust qos-preserving post-fault vnf placement," *IEEE Networking Letters*, 2023.

[5] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *8th IEEE International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2016, pp. 245–252.

[6] L. Askari, M. Tamizi, O. Ayoub, and M. Tornatore, "Protection strategies for dynamic vnf placement and service chaining," in *Proc. of the IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1–9.

[7] Y. Li, L. Li, J. Bai, X. Chang, Y. Yao, and P. Liu, "Availability and reliability of service function chain: A quantitative evaluation view," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 52, 2023.

[8] N. Shahriar, S. Taeb, S. R. Chowdhury, M. Zulfiqar, M. Tornatore, R. Boutaba, J. Mitra, and M. Hemmati, "Reliable slicing of 5g transport networks with bandwidth squeezing and multi-path provisioning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1418–1431, 2020.

[9] P. K. Thiruvasagam, A. Chakraborty, A. Mathew, and C. S. R. Murthy, "Reliable placement of service function chains and virtual monitoring functions with minimal cost in softwarized 5g networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1491–1507, 2021.

[10] R. Gour, V. Sikand, J. Chang, Z. Wu, G. Ishigaki, and J. P. Jue, "Traffic-weighted availability-guaranteed network slice composition with vnf replications," in *Proc. of IEEE International Conference on Communications (ICC)*, 2022, pp. 2278–2283.

[11] M. Nguyen, M. Dolati, and M. Ghaderi, "Deadline-aware sfc orchestration under demand uncertainty," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2275–2290, 2020.

[12] L. Wang, M. Dolati, and M. Ghaderi, "Change: Delay-aware service function chain orchestration at the edge," in *Proc. of IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*, 2021, pp. 19–28.
[13] X. Chen, J. Zhou, and S. Wei, "Sfc-ho: Reliable layered service function chaining," *IEEE Access*, vol. 10, pp. 106 352–106 368, 2022.
[14] D. Li, P. Hong, K. Xue, and J. Pei, "Availability aware vnf deployment in datacenter through shared redundancy and multi-tenancy," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1651–1664, 2019.
[15] J. Fan, M. Jiang, and C. Qiao, "Carrier-grade availability-aware mapping of service function chains with on-site backups," in *25th IEEE/ACM International Symposium on Quality of Service (IWQoS)*, 2017, pp. 1–10.
[16] M. Wang, B. Cheng, and J. Chen, "Joint availability guarantee and resource optimization of virtual network function placement in data center networks," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 821–834, 2020.
[17] P. M. Mohan and M. Gurusamy, "Resilient vnf placement for service chain embedding in diversified 5g network slices," in *Proc. of IEEE Conference on Global Communications (GLOBECOM)*, 2019, pp. 1–6.
[18] M. Wang, B. Cheng, S. Wang, and J. Chen, "Availability-and traffic-aware placement of parallelized sfc in data center networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 182–194, 2021.
[19] Y. Wu, W. Zheng, Y. Zhang, and J. Li, "Reliability-aware vnf placement using a probability-based approach," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2478–2491, 2021.
[20] L. Qu and C. Assi, "Reliability-aware multi-source multicast hybrid routing in softwarized networks," *IEEE Access*, vol. 8, pp. 113 331–113 341, 2020.
[21] M. Karimzadeh-Farshbafan, V. Shah-Mansouri, and D. Niyato, "Reliability aware service placement using a viterbi-based algorithm," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 622–636, 2020.
[22] Q. Zhang, F. Liu, and C. Zeng, "Adaptive interference-aware vnf placement for service-customized 5g network slices," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, 2019, pp. 2449–2457.
[23] C. Wang, Q. Hu, D. Yu, and X. Cheng, "Online learning for failure-aware edge backup of service function chains with the minimum latency," *IEEE/ACM Transactions on Networking*, 2023.
[24] Y. Liu, X. Shang, Y. Mao, Z. Liu, and Y. Yang, "Availability aware online virtual network function backup in edge environments," *IEEE Transactions on Mobile Computing*, 2023.
[25] M. Masoumi, I. de Miguel, R. J. D. Barroso, S. Hosseini, H. K. Janjua, N. Merayo, J. C. Aguado, and R. M. Lorenzo, "Efficient protected vnf placement and mec location selection for dynamic service provisioning in 5g networks," in *International Symposium on Distributed Computing and Artificial Intelligence*. Springer, 2023, pp. 448–456.
[26] J. Perez-Valero, A. Banchs, P. Serrano, J. Ortín, J. Garcia-Reinoso, and X. Costa-Pérez, "Energy-aware adaptive scaling of server farms for nfv with reliability requirements," *IEEE Transactions on Mobile Computing*, pp. 1–13, 2023.
[27] X. Chen, Z. Li, Y. Zhang, R. Long, H. Yu, X. Du, and M. Guizani, "Reinforcement learning–based qos/qoe-aware service function chaining in software-driven 5g slices," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 11, p. e3477, 2018.
[28] H. Hantouti, N. Benamar, M. Bagaa, and T. Taleb, "Symmetry-aware sfc framework for 5g networks," *IEEE Network*, vol. 35, no. 5, pp. 234–241, 2021.
[29] M. Liyanage, A. Braeken, P. Kumar, and M. Ylianttila, *IoT security: Advances in authentication*. John Wiley & Sons, 2020.
[30] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in *Proc. of Conference ACM SIGCOMM*, 2011, pp. 350–361.
[31] S. Hougardy, "The floyd–warshall algorithm on graphs with negative cycles," *Information Processing Letters*, vol. 110, no. 8-9, pp. 279–281, 2010.
[32] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *Proc. of 4th IEEE International Conference on Cloud Networking*, 2015, pp. 171–177.
[33] J. Fan, Z. Ye, C. Guan, X. Gao, K. Ren, and C. Qiao, "Grep: Guaranteeing reliability with enhanced protection in nfv," in *Proc. of ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, 2015, pp. 13–18.
[34] X. Shang, Y. Huang, Z. Liu, and Y. Yang, "Reducing the service function chain backup cost over the edge and cloud by a self-adapting scheme," *IEEE Transactions on Mobile Computing*, 2021.

Nadia Niknami received her B.S. degree in Computer Science from University of Isfahan, Iran, in 2011, and MSc degrees From Tarbiat Modares University, Tehran, Iran in 2015. She is currently pursuing the Ph.D. degree in the Department of Computer and Information Sciences, at Temple University, Philadelphia. Her current research focuses on cyber-security, attack-defense scenarios, Intrusion Detection Systems, and Neural networks.

Jie Wu is the Director of the Center for Networked Computing and Laura H. Carnell professor at Temple University. His current research interests include mobile computing and wireless networks, cloud computing, and network trust and security. Dr. Wu regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Service Computing, IEEE/ACM Transactions on Networking, and Journal of Computer Science and Technology. Dr. Wu is/was general chair/co-chair for IEEE DCOSS'09, IEEE ICDCS'13, ICPP'16, IEEE CNS'16, WiOpt'21, ICDCN'22, IEEE IPDPS'23, and ACM MobiHoc'23 as well as program chair/cochair for IEEE MASS'04, IEEE INFOCOM'11, CCF CNCC'13, and ICCCN'20. Dr. Wu is a Fellow of the AAAS and a Fellow of the IEEE.

Abdalaziz Sawwan is a fourth-year Ph.D. student in Computer and Information Sciences at Temple University. Sawwan received his bachelor's degree in Electrical Engineering from the University of Jordan in 2020 and his master's degree from Temple University in 2023. His current research interests include learning theory, multi-armed bandits, and wireless networks.