Hybrid Algorithm for VNF Hot Backup Migration in Follow-Me Mobile Edge Cloud

Yibo Wang and Junbin Liang *

Posted Date: 10 February 2025

doi: 10.20944/preprints202502.0698.v1

Keywords: Mobile edge cloud (MEC); virtual network function (VNF); hot backup; live Migration; latency; reliability

Article

Hybrid Algorithm for VNF Hot Backup Migration in Follow-Me Mobile Edge Cloud

Yibo Wang and Junbin Liang *

Guangxi Key Laboratory of Multimedia Communications and Network Technology, School of Computer and Electronics Information, Guangxi University, Nanning 530004, China
* Correspondence: liangjb@gxu.edu.cn

Abstract: In Follow-Me Mobile Edge Cloud environments, VNF instances dynamically move in tandem with user mobility. For latency-sensitive applications, hot backups aim to reduce service downtimes during primary VNF instance failures. However, as the distance between VNF instances and their hot backups shifts due to user mobility, recovery latency can sometimes exceed user expectations, leading to certain backups being perceived as unavailable. To maintain VNF reliability, it becomes essential to either deploy additional hot backups closer to the VNF instances or migrate the deemed unavailable backups to proximity, reinstating their usability. How to effectively leverage both the VNF and its failed hot backups to ensure VNF reliability, meet users' recovery latency demands, and minimize the overall cost of hot backup migration and redeployment is a challenging problem. This paper first formulates the problem as an integer linear programming problem and proves that the problem is NP-hard. Subsequently, we design an approximation algorithm for VNF backup deployment on edge servers. Theoretical analysis and simulation results demonstrate that this hybrid approach effectively reduces the cost of VNF backup migration.

Keywords: Mobile edge cloud (MEC); virtual network function (VNF); hot backup; live migration; latency; reliability

1. Introduction

Mobile edge computing (MEC) is a technology that provides cloud computing resources and services close to end-users and devices at the edge of the network[1]. The technology processes and stores data locally, enabling low-latency and high-bandwidth applications, rather than transmitting it to a central cloud server. MEC is overcoming traditional cloud computing's bottlenecks, including high network congestion and long communication delays. MEC facilitates new applications and services that require real-time, low-latency data processing, including smart cities, self-driving cars, and augmented reality. Currently, MEC adoption is rapidly accelerating in various industries, such as entertainment, transportation, and healthcare, and is playing a critical role in 5G network and Internet of Things (IoT) advancement[2].

The Follow-Me mobile edge computing (FEC) technique offers user mobility support[3]. As a user moves to different locations, the VNF migrates to the nearest edge node. Due to Follow-Me mobile edge computing moving computing and storage resources closer to users, it further reduces the distance and volume of data transmission, resulting in lower network costs. In this environments, where VNF instances dynamically migrate to maintain proximity to mobile users, hot backups play a critical role in minimizing service downtime for latency-sensitive applications during primary VNF failures. This approach ensures seamless service continuity through active synchronization and orchestration mechanisms.

The reliability of VNFs is crucial in the context of MEC, as it directly influences the continuity of services provided by MEC. In MEC scenarios, numerous critical applications and services depend on the normal operation of VNFs. The occurrence of VNF failures or unreliability can give rise to the following three issues: Firstly, it can result in service interruptions or performance degradation,

 doi:10.20944/preprints202502.0698.v1

manifesting as increased service response times, which can lead to user dissatisfaction. Secondly, it can cause service disruptions or erroneous data processing, thereby increasing the wastage of network resources. Thirdly, security functionalities such as firewalls and intrusion detection systems rely on reliable VNFs to provide protection. If VNFs have vulnerabilities or exhibit unreliability, it can elevate the risks of network attacks or data breaches.

To address these concerns, hot backups are often deployed in MEC environments to ensure high availability and minimize the impact of VNF failures. Hot backups maintain pre-instantiated VNF instances in standby mode, synchronized with the primary instance's state, enabling sub-second failover capabilities[4]. This failover capability is critical in MEC applications such as augmented reality or autonomous vehicle coordination, where even milliseconds of latency can have severe consequences. By leveraging hot backups, MEC systems can significantly improve service reliability, reduce latency, and ensure seamless operation of essential services.

Deploying multiple replicas of VNF is a common way to improve the reliability of VNF[5]. In the event of VNF failure, service requests automatically switch to an available VNF replica to maintain service continuity. As the user moves, VNF replicas are redeployed to the edge nodes close to the user, reducing bandwidth cost and minimizing service interruption time during potential VNF failures[6].

In the FMEC environment, user mobility introduces new complexities related to the deployment and migration of Virtual Network Function (VNF) backups. The unpredictable nature of user movement leads to dynamic changes in the latency between VNFs and their corresponding backups. When this latency exceeds a predefined threshold, it can make the backup function inaccessible or unreliable. As a result, when users move to a new region, both the VNFs providing the requested services and their corresponding Backup VNFs (BVNFs) must be either redeployed or migrated to locations that are closer to the user's new location, as depicted in Figure 1. However, frequent migration of VNF backups increases bandwidth consumption, thereby leading to substantial costs.

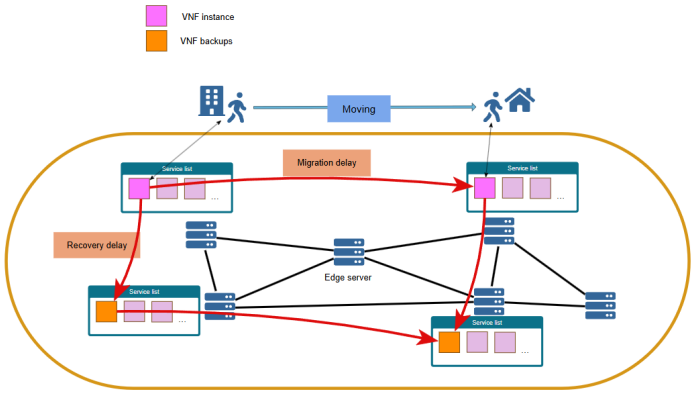


Figure 1. Delay-sensitive VNF migration.

In this paper, we demonstrate its NP-hardness by simplifying it into a multidimensional knapsack problem. Additionally, considering the perspectives of mobile users and edge servers, we jointly examine the impact of constraints such as latency, resource limitations, and VNF reliability on edge VNF backup migration, thus enhancing the accuracy of the backup migration cost model. The scale of the problem exponentially increases with the number of VNF backups that need to be migrated and the number of potential edge server locations for migration.

To resolve the above challenges, we propose a multi-stage approach. First, the state space is formed by considering the backup and potential migration locations for Virtual Network Functions (VNFs). Subsequently, various factors are taken into account to determine the most appropriate algorithm for reducing the state space, aiming to minimize the overall algorithm's runtime. Next, a thorough evaluation of each state's priority is conducted. During this process, factors such as resource requirements, reliability, migration's bandwidth cost, and latency before and after migration

 doi:10.20944/preprints202502.0698.v1

are considered in determining the priority of each state. Finally, based on the prioritization analysis, suitable states are selected for VNF backup migration.

The main contributions of this paper are as follows:

- To the best of our knowledge, we are the first to consider the secondary utilization of failed hot backups while deploying VNF backups. In doing so, we aim to minimize deployment and migration costs while taking into account factors such as latency and reliability. We model this problem as an integer linear programming problem and prove that it is NP-hard.
- To alleviate the solution complexity, we introduce a muti-stage scheme. This scheme can address the multi-source VNF hot backup migration problem with latency, capacity, and reliability constraints in polynomial time.
- Through extensive experiments, we validate that our proposed hybrid migration algorithm achieves the same level of user demand satisfaction as traditional migration algorithms while reducing VNF backup migration costs by approximately 15%.

2. Related Works

Virtualized Network Functions (VNFs) migration and backup are two critical aspects of Network Function Virtualization (NFV), playing an essential role in ensuring the seamless operation, high availability, and reliability of network services. Over the years, researchers and practitioners have devoted significant efforts to explore and improve VNF migration and backup mechanisms. In this section, we provide a comprehensive review of the existing literature, organizing the discussion into three main areas: VNF reliability, VNF migration approaches, and VNF backup strategies.

2.1. VNF Reliability

The issue of VNF reliability in the context of user mobility and recovery latency is a significant concern in telecommunications. As users move, the distance between VNF instances and their hot backups can increase, which may lead to recovery latencies that exceed user expectations. This situation can result in backups being perceived as unavailable, impacting the overall quality of service[7–10].

- **User Mobility:**The dynamic nature of user mobility necessitates that VNFs maintain low latency for recovery operations. When the distance between VNFs and their backups increases, the time taken to recover from failures can become unacceptable, leading to service disruptions.
- **Recovery Latency:**Recovery latency is critical in telecom services where outages must be minimized to milliseconds. If recovery times extend beyond this threshold, users may experience significant service degradation[11,12].
- **Backup Availability:**Hot backups must be readily accessible to ensure that if a primary VNF fails, service can be quickly restored without noticeable delays. However, as the distance increases, the effectiveness of these backups diminishes, making them seem unavailable when needed[13].

To address these challenges and maintain VNF reliability, several strategies can be employed:

- **Deploy Additional Hot Backups:** Placing more hot backup instances closer to active VNF instances can significantly reduce recovery times. This approach ensures that backups are within a minimal distance, facilitating quicker failover processes[14].
- **Migration of Backups:** Another effective strategy is to migrate deemed unavailable backups closer to their corresponding VNF instances. This proactive measure helps reinstate usability and minimizes latency during recovery operations[11,13].
- **Dynamic Resource Management:** Implementing dynamic resource management techniques allows for real-time adjustments based on user mobility patterns and network conditions. This includes optimizing the placement of both active and standby VNF instances to ensure they are strategically located for quick recovery[14].

 doi:10.20944/preprints202502.0698.v1

2.2. VNF Migration Approaches

VNF migration is a process of transferring a running VNF instance from one edge server to another without disrupting the service. Several migration approaches have been proposed to optimize VNF performance and resource utilization.

VNF migration can be categorized into offline migration and live migration. In offline migration, services are temporarily terminated during the migration process, while live migration[15–17] allows services to continue running without interruption for most of the migration process. Sun *et al.*[15] introduce improved serial migration strategies, a mixed migration strategy, and develops queuing models to quantify performance metrics. Rong *et al.*[16] classifies states into three types and uses three different techniques (warm-up, synchronization, and replay) for migration. Shi *et al.*[17]achieves live migration by continuously monitoring and dynamically selecting different models based on various memory/disk operations. Some articles utilize machine learning models to predict the probability of virtual network function failures and proactively initiate migration[17,18].

Cho *et al.*[19] proposed a novel VNF migration algorithm called VNF Real-time Migration (VNF-RM) aimed at reducing network latency during dynamic resource availability changes. They address the challenges of maintaining performance while migrating VNFs in real-time environments. Afrasiabi *et al.*[20] focused on optimizing cluster VNF migration while considering inter-VNF latency requirements. They provided insights into efficient migration strategies within service function chains. Wang *et al.*[21] proposed a method for coordinating updates to both the network function state and the software-defined networking (SDN) forwarding state during VNF migrations, ensuring consistency and reliability in service delivery. Tang *et al.*[22] presented a real-time VNF migration algorithm utilizing deep belief networks to predict future resource needs, enhancing the efficiency of migration processes in dynamic environments. Carpio *et al.*[23] discussed the flexibility and programmability offered by NFV, focusing on balancing the migration of VNFs while considering replication strategies to optimize resource usage and performance.

2.3. VNF Backup Strategies

VNF backup mechanisms play a crucial role in protecting critical data and configurations, enabling efficient recovery in the event of VNF failures or disasters[24–26].

Wang *et al.*[25]utilizes online learning techniques and the UCB (Upper Confidence Bound) algorithm to estimate the popularity and failure rate of Service Function Chaining (SFC). Subsequently, the RTSD (Real-Time Selection and Deployment) algorithm is employed to implement backup deployment. Shang *et al.*[26]extends the conventional static backup deployment by additionally considering dynamic backups. According to requirements, an online algorithm is utilized to dynamically adjust the quantity of dynamic backups. Iftikhar *et al.*[27] proposed a blockchain-based edge network authentication technology (BCAuthEN), which addresses security vulnerabilities in smart city networks through decentralized authentication, while ensuring continuous user verification and service availability in real-time edge networks. Ly *et al.*[28] provided a systematic literature review on the reliability of VNF chains, exploring the application of various algorithms and methods in ensuring reliability. The study covers a range of novel reliability algorithms and analyzes their applicability and effectiveness in modern networks. Yang *et al.*[29] studied the fault-tolerant placement problem of stateful virtual network functions, with the optimization goal of accommodating as many requests as possible while ensuring the continuity and stability of service chains.

Recognizing the importance of both VNF migration and backup, some studies have proposed integrated solutions that combine migration and backup mechanisms to achieve enhanced VNF resilience. Aidi *et al.*[30]leverage VNF migration to relocate the VNFs in order to minimize the number of shared VNF backups.

Some studies also employ deep learning techniques to ensure the recovery of failed VNFs.Ishigaki *et al.*[31] address the progressive recovery problem in virtualized networks with limited resources, where interdependencies between virtual network functions (VNFs) and infrastructure elements

 doi:10.20944/preprints202502.0698.v1

complicate the recovery process. They propose DeepPR, a Deep Reinforcement Learning-based technique, which achieves near-optimal solutions and outperforms baseline heuristics in terms of robustness to adversarial failures.Qu *et al.*[32] propose a priority-awareness deep reinforcement learning (PA-DRL) algorithm to enhance the reliability and optimization of service function chains (SFC) by dynamically determining backup server placement. Mao *et al.*[33] propose DDQP, a deep reinforcement learning-based online SFC placement method that integrates active and standby VNF instances to enhance fault tolerance and service reliability. Ros*et al.*[34] propose a graph neural network-based deep reinforcement learning (GRL-SFT) model to enhance the fault tolerance of service function chains (SFC) in NFV environments. Wang*et al.*[35] propose a deep reinforcement learning-based joint VNF partition and hybrid backup scheme for VNF orchestration, enhancing the reliability and delay performance of network slices at minimal cost.

However, the aforementioned articles do not consider utilizing previously deployed VNF backups when discussing VNF backups. Specifically, after each VNF migration, various VNF backup methods are employed to redeploy backups around the migrated VNF to reduce backup recovery latency. However, the migration cost of existing VNF backups may be lower than the cost of redeploying VNF backups. In contrast, in our approach, during the VNF migration process, we utilize existing VNF backups. To minimize costs, we only perform live migration of appropriate VNF backups if the backup recovery time exceeds user requirements, rather than redeploying all VNF backups.

As shown in Figure 2, suppose the VNF instance requested by the user has a backup. When the VNF instance migrates from A to B along with the user's movement, the VNF backup on edge server C becomes unavailable due to excessive recovery latency. In this case, the VNF backup needs to be redeployed on D. If the migration cost from L3 is less than L4, the VNF backup migration algorithm can be used to migrate the backup from C to D. Otherwise, the backup should be redeployed from B to D.

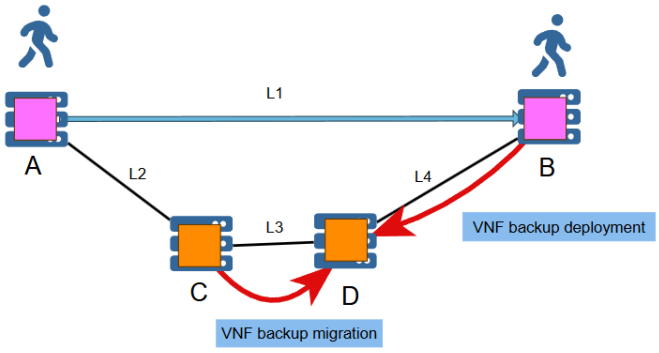


Figure 2. Migration or redeployment of VNF backup.

3. System Model and Problem Formulation

3.1. Network Model

In this paper, we examine the Follow-me MEC system with limited resources and represent it as an undirected weighted graph $G = (V, E)$. The set V represents edge servers in the mobile edge network, and the set $E = e_{u,v}(u, v \in V)$ signifies links between two servers u and v in G.

3.2. User Requests

We use $r = (F, T^{err}, C)$ to denote a single user request, and $R = r_m$ to denote the set of requests. For simplicity, we assume that each user requests a VNF instance, and this can be extended to multiple VNF instances. $F = f_i(0 \leq i \leq |F|)$ denotes the VNF instance and its backups requested by a user. Here, f_0 represents the user's requested VNF instance, and $f_i(1 \leq i \leq |F|)$ represents backup instances of the user's requested VNF instance. The edge server's location of the backup VNF instance f_i is denoted by V_{f_i}. The parameter T^{err} defines the maximum average fault recovery delay that the user

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 February 2025 doi:10.20944/preprints202502.0698.v1

can tolerate. In this study, we consider real-time backups and assume that the VNFs and their backups utilize the same computing resources, denoted by C.

3.3. Delay Constraint

Our study uses a binary variable, denoted by P_{f_i,v_n,r_m}, to identify the placement of VNF backups.

$$P_{f_i,v_n,r_m} = \begin{cases} 1, & \text{if VNF } f_i \in r_m \text{ is deployed on } v_n \\ 0, & \text{otherwise} \end{cases} \quad \forall f_i \in F, \forall v_n \in V, \forall r_m \in R \tag{1}$$

The value of P_{f_i,v_n,r_m} equals 1 if VNF f_i of request r_m is hosted on edge server v_n, and it equals 0 otherwise.

Storing VNF backups on two separate servers can mitigate single-point failures, thereby improving system availability. In the event of a server failure, the backup system can continue to function without interruption and rapidly resume control of the primary instance, thereby ensuring the VNF system remains fully operational.

$$P_{f_i,v_n,r_m} + P_{f_j,v_n,r_m} = 1, \forall f_i, f_j \in F, i \neq j, \forall v_n \in V, \forall r_m \in R \tag{2}$$

$$\sum_{\forall v_n \in V, \forall r_m \in R} P_{f_i,v_n,r_m} = 1, \forall f_i \in F \tag{3}$$

Assuming that the delay of the link (u, v) between edge servers u and v is relatively stable, we employ the Dijkstra algorithm to determine the shortest delay between any two edge servers in an undirected weighted graph. We denote the shortest delay between edge servers (u, v) as $l(u, v)$, and represent the set of link delays as $L = l(u, v) \mid (u, v) \in V$. It is worth noting that we use $l(V_{f_0}, V_{f_i})$ to calculate the necessary delay required for recovery from each backup of VNF f_i in the event of a failure.

Hence, the formula for determining the average delay in recovering from a fault when the VNF instance f_0 fails is obtainable and given as follows:

$$T_{r_m}^{all} = \frac{\sum_{i=1}^{|F|} l(V_{f_0}, V_{f_i})}{|F|} \tag{4}$$

Therefore, the delay constraint that we need to satisfy is:

$$T_{r_m}^{all} \leq T^{err} \tag{5}$$

3.4. Resource Constraint

The resource constraints of the edge server must be taken into consideration for successful deployment of VNF and its backups.To elaborate, the total resource requirements of VNF instances and their backups should not surpass the available resources cap_n on the edge server n.

$$\sum_{r_m \in R} \sum_{f_i \in F} P_{f_i,v_n,r_m} \times C \leq cap_n, \forall v_n \in V \tag{6}$$

3.5. Migration Cost Model

When migrating the backup of a VNF from server u to server v, a transmission cost is incurred, which depends on the link latency $l(u, v)$ between servers u and v. This cost is defined as a generally non-decreasing function $c(x)$, where $x = l(u, v)$. Since link latency is typically a function of the distance between two nodes, it increases with distance.

 doi:10.20944/preprints202502.0698.v1

we define the transmission cost functions $c(x)$ in a constant-plus-exponential form. This form of cost function can approximate many real-world scenarios by changing the parameters:

$$c(x) = \alpha_a + \alpha_b \alpha_c^{(x)}, x \geq 0 \tag{7}$$

The constants α_a, α_b, and α_c are fixed values in the model that can be modified based on specific scenarios.

3.6. Reliability Reward Model

To quantify the contribution of VNF backups to VNF reliability, we first calculate the reliability of a VNF instance using the following formula:

$$y = \prod_{f_i \in F} \prod_{v_n \in V} (1 - p_{f_i})(1 - p_{v_n}) P_{f_i, v_n, r_m} \tag{8}$$

Where p_{f_i} denotes the failure probability of VNF f_i, and p_{v_n} denotes the failure probability of edge server v_n. The failure probability p is calculated as follows:

$$p = \frac{MTTR}{MTTN + MTTR} \tag{9}$$

$MTTN$ and $MTTR$ refer to the average time to reach normal operation and the average time to repair, respectively.

When a VNF deploys a backup to an edge server node, it can provide reliability assurance, increasing the VNF's revenue. This revenue is related to the reliability of the backup and can be defined as a non-decreasing function $d(y)$. It's essential that the backup is readily available to be used when necessary, as downtimes can severely impact the revenue and user satisfaction.

we define the reliability reward functions $d(y)$ in a constant-plus-exponential form.

$$d(y) = \beta_a + \beta_b \beta_c^{(y)} \tag{10}$$

The constants β_a, β_b, and β_c are fixed values in the model that can be modified based on specific scenarios.

3.7. Problem Definition

Therefore, we define the migration cost for VNF backup f_i when it migrates from edge server u to edge server v as:

$$C_{f_i}^{mig}(u, v) = \gamma_{f_i} \times c(l(u, v)) - \delta_{f_i} \times (d(v) - d(u)) \tag{11}$$

γ_{f_i} and δ_{f_i} are real-valued parameters. They can be modified according to the user's emphasis on VNF reliability and transmission cost, in order to adapt to different situations.

According to the above description, our problem is to migrate suitable VNF backups to suitable edge servers, while considering the constraints of average failure latency and edge server resource, and achieve the minimization of the total migration cost with consideration of reliability gains. We can get the VNF backup migration problem formulated as:

Objective :

$$\min \sum_{r_m \in R} \sum_{i \in F} C_{f_i}^{mig}(u, v) \tag{12}$$

Subject to :

$$C_{f_i}^{mig}(u, v) = \gamma_{f_i} \times c(l(u, v)) - \delta_{f_i} \times (d(v) - d(u)) \tag{13}$$

$$c(x) = \alpha_a + \alpha_b \alpha_c^{(x)}, x \geq 0 \tag{14}$$

$$d(y) = \beta_a + \beta_b \beta_c^{(y)} \tag{15}$$

 doi:10.20944/preprints202502.0698.v1

$$y = \prod_{f_i \in F} \prod_{v_n \in V} (1 - p_{f_i})(1 - p_{v_n}) P_{f_i, v_n, r_m} \tag{16}$$

$$\sum_{r_m \in R} \sum_{f_i \in F} P_{f_i, v_n, r_m} \times C \leq cap_n, \forall v_n \in V \tag{17}$$

$$P_{f_i, v_n, r_m} + P_{f_j, v_n, r_m} = 1, \forall f_i, f_j \in F, i \neq j, \forall v_n \in V, \forall r_m \in R \tag{18}$$

$$\sum_{\forall v_n \in V, \forall r_m \in R} P_{f_i, v_n, r_m} = 1, \forall f_i \in F \tag{19}$$

3.8. NP-Hardness of VNF Backup Migration Problems

Theorem 1. *VNF backup migration problem defined in (12) is an NP-hard problem*

Proof. We demonstrate the NP-hardness of VNF backup migration by reducing it to the multidimensional knapsack problem, which is a well-established and notorious NP-hard problem with the following definition.The goal of this problem is to select n items from a set of items $Item = (i_1, i_2, ..., i_m)$, where each item has a unique weight $weight_i$, and value $value_i$. Choose n items and place them into a backpack with limited capacity cap, so as to maximize the total value of items while following the constraint that the total weight cannot surpass the backpack's capacity cap.

We can consider the current position u_i and the possible edge server v_i that each backup i can migrate to as an item, with a total of $\sum_{i \in n} u_i \times v_i$ items. The time delay T_v from the primary VNF to the edge server after backup migration and the total number of backups n that need to be migrated must satisfy the constraint that the total latency from the primary VNF to all n backups cannot exceed T_{all}. The reliability value $d(v)$ of each backup i minus the migration cost from u_i to v_i can be considered as the value of the item. It is NP-hard to maximize the total value of the selected n items while satisfying the above constraints. In this article, we also need to consider the resource constraints of the edge servers for the VNF backup migration problem, and each BVNF can only and must be selected once, which imposes additional constraints on the selection of items.Therefore, the problem of VNF backup migration is an NP-hard problem[36]. □

4. Method

In this section, to tackle the issue of VNF backup migration, we first conduct a more in-depth analysis in Section IV-A. Subsequently, in Section IV-B, we present corresponding algorithms to address the existing challenges.

4.1. Problem Analysis

We denote the possible edge server v_i that backup i can migrate to and its current position u_i as $s_i = (u_i, v_i)$.However, this further increases the state space s_i, and we need to simplify the state space to avoid excessively long solving time for this problem. When migrating, we prioritize avoiding migration to edge servers with existing BVNF and locations farther away from the main VNF instance than the current location. The VNF backup selection and migration problem can be decomposed into the following steps:

1. Preprocessing is performed on the graph within the network model, and the preprocessed network graph can be utilized multiple times before any changes occur in network nodes. This efficiently decreases the execution time of the algorithm.
2. Anticipating whether VNF migration caused by user movement will result in exceeding latency limits, preparations are made for VNF backup migration if the latency constraints are exceeded.
3. When VNF backup migration is required, preliminary filtering of potential migration locations is conducted based on specified constraints. This process reduces the algorithm's runtime by narrowing down the state space.

 doi:10.20944/preprints202502.0698.v1

4. The optimized Kuhn-Munkres algorithm is employed to solve the combinatorial optimization problem. The process involves sequentially computing whether the constraints are satisfied after VNF backup migration based on priority.

Throughout this process, we encounter the following challenges: the reliability benefits of dynamic VNF backup, the fault latency of dynamic VNF backup, and the finite resources of edge servers.

To address the challenges mentioned above, we will provide a detailed discussion of our approach and the general process of VNF backup migration in Section IV-B. This aims to identify a suitable set of VNF backups for edge migration.

4.2. VNF Backup Migration Algorithm

To address the issue mentioned above, We propose a VNF backup Migration(VBM) algorithm, by jointly considering reliability, limited capacity and latency. As VNFs migrate with users, the fault recovery latency of VNF backups varies. VBM reduces fault recovery latency by migrating suitable VNF backups. Hence, it avoids the costs associated with frequent redeployment of all VNF backups.

We propose an algorithm named VNF Backup Migration Algorithm based on the above ideas.

Algorithm 1: VNF Backup Migration Algorithm

Input: $G = (V, E)$, R
Output: S

1 Preprocess and save the network model graph G using Dijkstra's algorithm.
2 **for** $r \in R$ **do**
3 Calculate afl_{mig} after VNF migration.
4 **if** $afl_{mig} > afl_r$ **then**
5 **for** $u \in V$ **do**
6 **if** $e_{aim,u} < afl_{max}$ **then**
7 $|BVNF_{loc}^{mig}| \leftarrow u$.
8 **end**
9 **end**
10 $||W_G|| \leftarrow |BVNF_{loc}^{mig}|$.
11 Generate a set of migratable options S_{opt}^{mig} using Kuhn-Munkres algorithm.
12 Sort the migratable options S_{opt}^{mig} by their priorities.
13 **while** $T^{err} \leq T_{r_m}^{all}$ **do**
14 $S \leftarrow$ Select highest priority option $s(s \in S_{opt}^{mig})$ Update $T_{r_m}^{all}$
15 **end**
16 **end**
17 return S
18 **end**

In Algorithm 1, we first preprocess the network graph G by employing the Dijkstra algorithm to determine the shortest distances between all nodes in G (Line 1). For each mobile user's request r, we calculate the average fault recovery latency of the VNF after the user migration afl_{mig} (Line 3). If afl_{mig} is greater than the average fault recovery latency that the user can tolerate, we optimize by migrating VNF backups (Line 4-16). Based on the edge weights in our network model G, we first filter the nodes eligible for VNF backup migration to prevent an excessive number of nodes from negatively affecting the algorithm's performance (Line 5-9). Next, we generate a weight matrix based on the current positions of the BVNFs and their potential migration locations (Line 10). We apply the improved Kuhn-Munkres algorithm to solve the minimum weight matching problem, thereby obtaining the feasible VNF backup migration options S_{opt}^{mig} (Line 11). For each S_{opt}^{mig}, we calculate its priority and subsequently sort the values (Line 12). We calculate the average fault latency after the

 doi:10.20944/preprints202502.0698.v1

migration of request $T_{r_m}^{all}$ and compare it with the maximum acceptable average fault latency for users T^{err}. When $T_{r_m}^{all}$ exceeds the threshold T^{err} that users can accept, we select the highest-priority option from the S_{opt}^{mig} for migration. We then recalculate $T_{r_m}^{all}$ after the migration, continuing this process until it meets user requirements(Line 13-15).

As described above, we designed a VNF backup migration algorithm inspired by the Kuhn-Munkres algorithm. This algorithm aims to meet user latency requirements while minimizing the costs associated with VNF backup migration. According to Theorem 1, VNF backup migration is NP-hard, making it difficult to find an optimal solution within polynomial time.

Therefore,in Algorithm 1, we employ the improved Kuhn-Munkres algorithm to solve the basic minimum weight matching problem. This serves as a local solution for the network graph G, which has been preprocessed using the Dijkstra algorithm. The time complexity of this approach is $O(k^3)$, where k represents the number of VNF backups.

4.3. Optimized VNF Backup Algorithm

In Algorithm 2, we introduce a hybrid VNF backup algorithm that integrates Algorithm 1 with the traditional VNF re-deployment method to enhance adaptability across different network conditions. The motivation for this approach arises from the observation that Algorithm 1, while effective in many scenarios, does not always outperform traditional VNF re-deployment. By dynamically selecting the optimal strategy, Algorithm 2 ensures lower migration costs and better overall performance.

Algorithm 2: Hybrid VNF Backup Algorithm

Input: $G = (V, E)$, R
Output: S

```
1  Preprocess and save the network model graph G using Dijkstra's algorithm.
2  for r ∈ R do
3      Calculate VNF backup migration cost C^mig using Algorithm 1.
4      Calculate VNF re-deployment cost C^redeploy using traditional VNF re-deployment
         algorithm.
5      if C^mig ≤ C^redeploy then
6          Use Algorithm 1 for VNF backup migration and obtain solution S^mig.
7          S ← S^miga.
8      end
9      else
10         Use traditional VNF re-deployment algorithm and obtain solution S^redeploy.
11         S ← S^redeploy.
12     end
13 end
14 return S.
```

We begin by preprocessing the network graph G using Dijkstra's algorithm to determine the shortest distances between all nodes in G (Line 1). For each mobile user request r, we calculate the VNF backup migration cost C^{mig} based on Algorithm 1 (Line 3). Simultaneously, we compute the cost of re-deploying the VNF using a traditional re-deployment algorithm $C^{redeploy}$ (Line 4). The objective is to determine which approach incurs the lowest cost while maintaining acceptable latency constraints.

If the migration cost using Algorithm 1 is lower ($C^{mig} \leq C^{redeploy}$), we select Algorithm 1 for VNF backup migration (Lines 5-7). This process follows the Kuhn-Munkres-based optimization framework outlined in Algorithm 1, where a bipartite graph matching strategy is employed to find the optimal VNF backup placement while minimizing migration cost.

Conversely, if the traditional VNF re-deployment cost is lower ($C^{redeploy} < C^{mig}$), we utilize the traditional method for backup re-deployment (Lines 9-11). This typically involves deploying a

 doi:10.20944/preprints202502.0698.v1

new instance of the VNF at a location that minimizes operational cost while satisfying user latency constraints.

The proposed hybrid algorithm dynamically adapts to different network conditions by selecting the optimal migration strategy based on cost minimization. Compared to a static backup migration approach, Algorithm 2 offers greater flexibility and broader applicability across diverse network environments.

Algorithm 2, by combining Algorithm 1 with traditional redeployment techniques, not only ensures that the user's recovery latency requirements are met but also further reduces the total cost of VNF backup by dynamically selecting the migration strategy.

5. Experiment

5.1. Experimental Setup

This study evaluates the performance of the proposed VNF (Virtual Network Function) backup migration algorithm through extensive simulation experiments. All simulations were conducted on a laptop running the Windows 11 operating system, equipped with an AMD Ryzen 7 5800H CPU @ 3.20GHz and 16GB RAM. Python and the NetworkX library were used to construct and simulate the edge network environment.

To comprehensively assess the algorithm's performance across different environments, we defined a set of adjustable variables. Among them, the following are critical:

1. **Number of Edge Server Nodes:** This variable simulates edge cloud environments of various sizes, ranging from 20 to 200 nodes. When the number of edge server nodes is large, the algorithm optimizes by filtering out nodes that are too far from the VNF instances.
2. **Number of VNF Backups:** This variable can be adjusted according to application scenario requirements to improve the reliability of VNFs by increasing the number of backups.
3. **Graph Sparsity:** The sparsity of the network is controlled by adjusting the probability of link existence between edge nodes (link probability ranging from 0.1 to 0.5).
4. **Link Latency:** The latency of the link between two directly connected servers is a random integer between 1 and 20, simulating the uneven transmission delays typically encountered in real networks.

In addition, other relevant variables were set to simulate different environments and requirements to further test the algorithm's performance. These adjustable variables increase the flexibility of the experiments and allow for verification of the algorithm's adaptability across a broader range of application scenarios. The generated edge network is modeled as an undirected connected graph to ensure that there is a reachable path between all server nodes. To reduce the randomness introduced by the random generation of network structures, each experimental setup was run multiple times, and the average results were taken.

5.2. Evaluation of the Superiority of Our Algorithm

To validate the superiority of the proposed VNF backup migration algorithm, we designed a series of comparative experiments. As discussed earlier, our algorithm primarily addresses several key challenges related to VNF backup in edge environments: the reliability of VNFs, the cost of VNF backup deployment or migration, and the recovery delay of VNF backups. In this section, we designed three baseline schemes for comparison to further assess the performance of our algorithm. The specific comparison schemes are as follows:

5.2.1. Baseline Schemes

To evaluate the performance of our algorithm, we designed three baseline schemes:

- **Baseline Scheme 1:** This scheme uses a VNF backup migration strategy but does not optimize migration costs. The focus is on meeting user latency constraints, but it may result in high migration overhead.

 doi:10.20944/preprints202502.0698.v1

- **Baseline Scheme 2 (Traditional Scheme):** This is a conventional VNF migration method. After migrating the VNF instance, a hot backup is redeployed on a nearby node to ensure service reliability.
- **Baseline Scheme 3:** This scheme employs a VNF backup migration strategy while optimizing migration costs to reduce the overhead associated with backup deployment. However, in some scenarios with high user latency constraints, it may not meet the latency requirements of user requests.

5.2.2. Comparison with the Proposed Algorithm

Building upon Baseline Scheme 3, we further combine the advantages of Baseline Schemes 2 and 3. The aim is to meet user latency demands while further reducing the overall cost of VNF backup migration. To evaluate its performance, we compare it against the aforementioned three baseline schemes.

5.2.3. Experimental Methodology and Evaluation Metrics

All schemes are evaluated under the same experimental conditions, and the main evaluation metrics include:

- **Migration Cost:** This metric assesses the overall resource consumption required to meet the user's reliability and backup recovery delay requirements after VNF instance migration. Different environmental parameters may affect this cost.
- **User Latency Satisfaction Rate:** This measures the proportion of successful service recovery within the user's requested latency when a VNF instance fails.
- **Backup Migration Duration:** This metric measures the time taken for VNF backup migration to complete after the primary VNF instance migration.

5.3. Experimental Results and Analysis

To comprehensively evaluate the performance of the proposed hybrid migration algorithm, we conducted in-depth experimental analyses across four control variables: the number of VNF backups, the number of user requests, the number of edge network nodes, and the connectivity of the network graph. In the experiments, we compared the three baseline algorithms (Baseline Scheme 1, Baseline Scheme 2, Baseline Scheme 3) with our hybrid VNF backup migration algorithm, considering multiple evaluation metrics such as recovery delay, migration cost, and user latency satisfaction rate, to investigate the algorithm's applicability in different environments. The data presented in the following analysis were derived from 1000 randomly generated user requests.

5.3.1. Impact of Number of VNF Backups

We first evaluated the impact of varying the number of VNF backups on the performance of the different schemes. The number of backups was gradually increased from 5 to 20, with an interval of 5.

- **Applicability Ratio:** The applicability ratio refers to the proportion of user requests that can be satisfied using the corresponding algorithm, given a latency threshold. As shown in Figure 3, for different numbers of VNF backups, Baseline Scheme 3, despite having the lowest VNF backup migration cost, only applies to a small portion of user requests. In contrast, our proposed hybrid backup migration algorithm achieves an applicability ratio comparable to that of Baseline Scheme 2.
- **Migration Cost Savings Ratio:** The migration cost savings ratio refers to the proportion of migration costs saved when using the hybrid VNF backup migration algorithm, compared to Baseline Scheme 2, for a given user request. As shown in Figure 4, the savings ratio decreases as the number of VNF backups increases.

 doi:10.20944/preprints202502.0698.v1

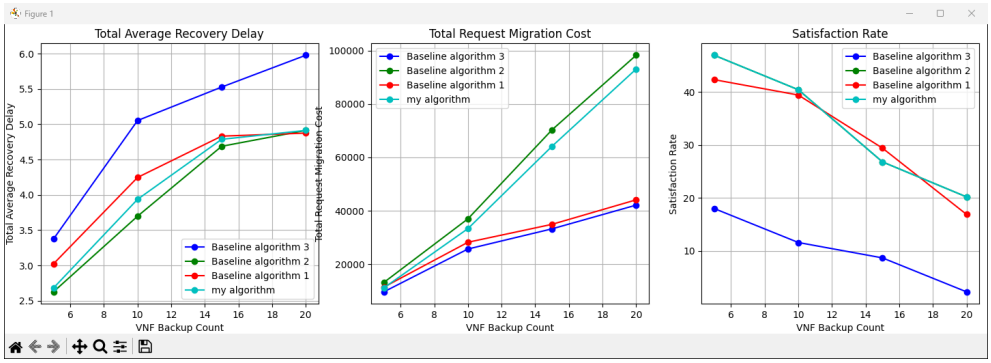


Figure 3. Impact of the number of VNF backups.

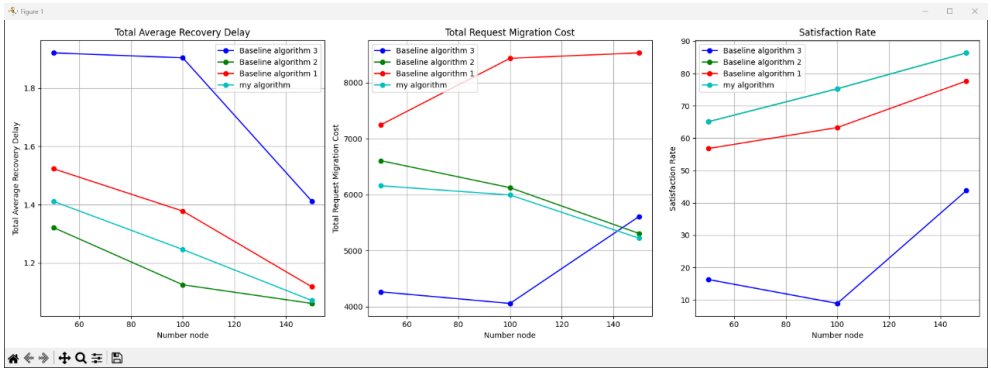


Figure 4. Impact of the number of edge nodes.

5.3.2. Impact of Number of Edge Network Nodes

To assess the impact of the scale of the edge network on the algorithm's performance, we adjusted the number of network nodes, increasing from 50 to 200 with an interval of 50.

- **Applicability Ratio:** As shown in Figure 5, although the overall applicability ratio for all algorithms increases significantly with the number of edge network nodes, the applicability ratio of Baseline Scheme 3 remains lower than that of the other algorithms. The hybrid VNF backup migration algorithm achieves the same applicability rate as Baseline Scheme 2, outperforming the other baseline schemes.
- **Migration Cost Savings Ratio:** As shown in Figure 6a, the migration cost savings ratio decreases as the number of edge network nodes increases.

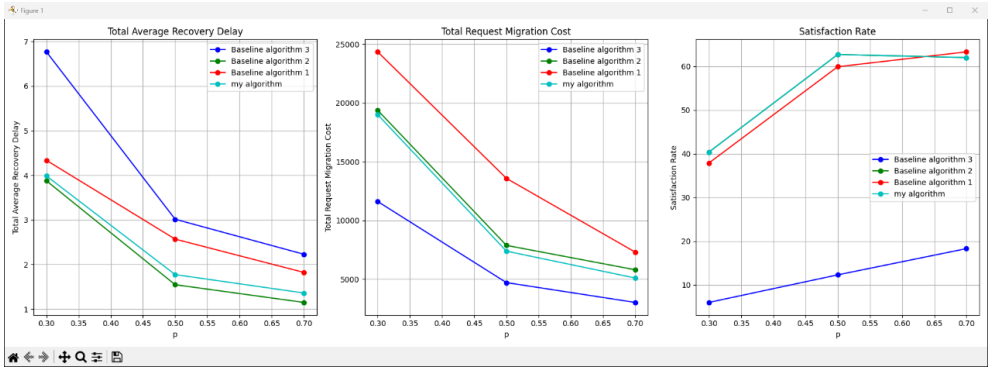
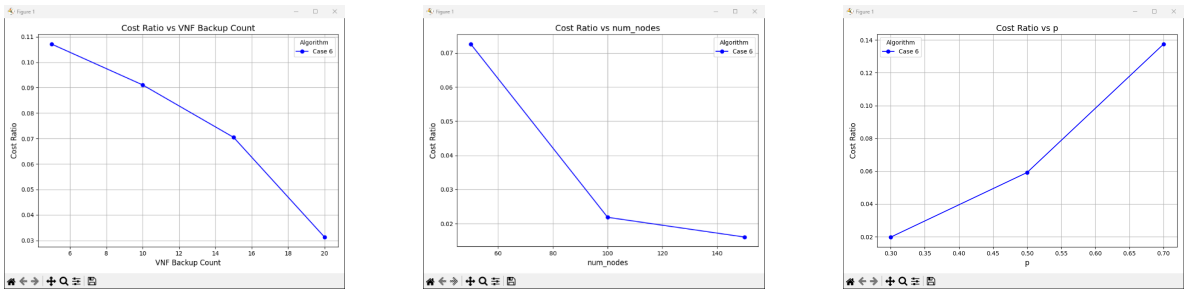


Figure 5. Impact of the number of edge nodes.

 doi:10.20944/preprints202502.0698.v1



(**a**) (**b**) (**c**)

Figure 6. Impact of different environments on algorithm performance: (**a**) Number of VNF backup. (**b**) Number of edge nodes within the region. (**c**) Connectivity of the network graph G.

5.3.3. Impact of Network Graph Connectivity

Finally, we analyzed the impact of network graph connectivity on algorithm performance. The connectivity was varied from 0.1 (sparse graph) to 0.7 (high connectivity graph), with an interval of 0.2.

- **Applicability Ratio:** As shown in Figure 6b, the applicability ratio of all algorithms increases with the network graph's connectivity.
- **Migration Cost Savings Ratio:** As shown in Figure 6c, the migration cost savings ratio increases significantly as the network graph connectivity improves.

5.4. Summary of Superiority

The experimental results demonstrate that the proposed hybrid VNF backup migration algorithm achieves the same level of user latency satisfaction as Baseline Scheme 2, while offering lower backup migration costs. The optimization ratio of backup migration costs varies with the number of backups, network graph connectivity, and the number of edge nodes. The optimization ratio is higher in small-scale, high-connectivity edge networks, or when there are fewer VNF backups, reaching up to approximately 15%.

6. Conclusions

In this paper, we investigated the problem of hot backup migration for latency-sensitive Virtual Network Functions (VNFs) in Follow-Me Mobile Edge Cloud (FMEC) environments. We formulated the problem as an integer linear programming model and proved its NP-hardness. To address the complexity of the problem, we proposed a hybrid VNF backup migration algorithm that jointly considers latency constraints, resource limitations, and reliability requirements. Our algorithm dynamically selects between backup migration and re-deployment strategies to minimize overall migration costs while ensuring service continuity. Theoretical analysis and simulation results demonstrate that our approach significantly reduces the cost of backup migration while maintaining user latency satisfaction at a level comparable to traditional backup deployment methods. Future research directions include integrating machine learning techniques to enhance predictive migration decisions and extending the model to multi-tenant edge computing scenarios.

Author Contributions: Conceptualization, Y.W. and J.L.; methodology, Y.W.; software, Y.W.; validation, Y.W.; formal analysis, Y.W. and J.L.; investigation, Y.W.; resources, Y.W.; data curation, Y.W.; writing—original draft preparation, Y.W.; writing—review and editing, Y.W. and J.L.; visualization, Y.W.; supervision, Y.W. and J.L.; project administration, Y.W. and J.L.; funding acquisition, Y.W. and J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported/funded by the National Natural Science Foundation of China (Grant No. 62362005).

Institutional Review Board Statement: Not applicable.

 doi:10.20944/preprints202502.0698.v1

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available in Github at https://github.com/shana0325/VNF.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

G	Undirected weighted graph representing the mobile edge network.
V	The set of edge servers in the mobile edge network
E	The set signifies links$\{e_{u,v}(u,v \in V)\}$
R	The set of users request $R = \{r_1, r_2, \ldots, r_n\}$
F	The VNF instance and its backups requested by a user.$F = f_i(0 \leq i \leq \|F\|)$
V_{f_i}	The edge server's location of the backup VNF instance f_i
T^{err}	Maximum acceptable average fault latency for users.
C	Computational resources required for VNF and its backup.
P_{f_i,v_n,r_m}	{0,1}, Used to indicate the location of the VNF.
L	The set of link latency between edge servers$\{l_{u,v}(u,v \in V)\}$
$T^{all}_{r_m}$	Average fault latency after request r_m migration.
cap_n	Available resources on edge server n before migration.
$c(x)$	Migration cost for VNF and its backup.
$d(y)$	The reliability of the backup
$C^{mig}_{f_i}(u,v)$	The migration cost of VNF f_i from edge server u to v
afl_{mig}	Average fault latency after VNF migration.
S	The set of optional migration states.$\{s(u,v)(u,v \in V)\}$
$s_{prio}(u,v)$	Priority of migration states.$\{s_{prio}(u,v)(u,v \in V)\}$
S^{mig}_{opt}	The set of migratable options.

References

1. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Communications Surveys & Tutorials* **2017**, *19*, 2322–2358. https://doi.org/10.1109/COMST.2017.2745201.
2. Taleb, T.; Samdanis, K.; Mada, B.; Flinck, H.; Dutta, S.; Sabella, D. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Communications Surveys & Tutorials* **2017**, *19*, 1657–1681. https://doi.org/10.1109/COMST.2017.2705720.
3. Ouyang, T.; Zhou, Z.; Chen, X. Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing. *IEEE Journal on Selected Areas in Communications* **2018**, *36*, 2333–2345. https://doi.org/10.1109/JSAC.2018.2869954.
4. ETSI, G. Mobile edge computing (mec); deployment of mobile edge computing in an nfv environment. *ETSI ISG* **2018**.
5. Liang, W.; Ma, Y.; Xu, W.; Xu, Z.; Jia, X.; Zhou, W. Request Reliability Augmentation With Service Function Chain Requirements in Mobile Edge Computing. *IEEE Transactions on Mobile Computing* **2022**, *21*, 4541–4554. https://doi.org/10.1109/TMC.2021.3081681.
6. Shang, X.; Huang, Y.; Liu, Z.; Yang, Y. Reducing the Service Function Chain Backup Cost Over the Edge and Cloud by a Self-Adapting Scheme. *IEEE Transactions on Mobile Computing* **2022**, *21*, 2994–3008. https://doi.org/10.1109/TMC.2020.3048885.
7. Li, J.; Guo, S.; Liang, W.; Chen, Q.; Xu, Z.; Xu, W.; Zomaya, A.Y. Digital Twin-Assisted, SFC-Enabled Service Provisioning in Mobile Edge Computing. *IEEE Transactions on Mobile Computing* **2024**, *23*, 393–408. https://doi.org/10.1109/TMC.2022.3227248.
8. Németh, B.; Molner, N.; Martín-Pérez, J.; Bernardos, C.J.; de la Oliva, A.; Sonkoly, B. Delay and Reliability-Constrained VNF Placement on Mobile and Volatile 5G Infrastructure. *IEEE Transactions on Mobile Computing* **2022**, *21*, 3150–3162. https://doi.org/10.1109/TMC.2021.3055426.

 doi:10.20944/preprints202502.0698.v1

9. Wu, Y.; Zheng, W.; Zhang, Y.; Li, J. Reliability-Aware VNF Placement Using a Probability-Based Approach. *IEEE Transactions on Network and Service Management* **2021**, *18*, 2478–2491. https://doi.org/10.1109/TNSM.2021.3093199.
10. Ghazizadeh, A.; Akbari, B.; Tajiki, M.M. Joint Reliability-aware and Cost Efficient Path Allocation and VNF Placement using Sharing Scheme, 2020, [arXiv:cs.NI/1912.06742].
11. Fang, J.; Zhao, G.; Xu, H.; Tu, H.; Wang, H. Reveal: Robustness-aware VNF placement and request scheduling in edge clouds. *Computer Networks* **2023**, *233*, 109882. https://doi.org/https://doi.org/10.1016/j.comnet.2023.109882.
12. Jin, P.; Fei, X.; Zhang, Q.; Liu, F.; Li, B. Latency-aware VNF Chain Deployment with Efficient Resource Reuse at Network Edge. In Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, 2020, pp. 267–276. https://doi.org/10.1109/INFOCOM41043.2020.9155345.
13. Yuan, G.; Xu, Z.; Yang, B.; Liang, W.; Chai, W.K.; Tuncer, D.; Galis, A.; Pavlou, G.; Wu, G. Fault tolerant placement of stateful VNFs and dynamic fault recovery in cloud networks. *Computer Networks* **2020**, *166*, 106953. https://doi.org/https://doi.org/10.1016/j.comnet.2019.106953.
14. Wang, C.; Hu, Q.; Yu, D.; Cheng, X. Proactive Deployment of Chain-based VNF Backup at the Edge using Online Bandit Learning. In Proceedings of the 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), 2021, pp. 740–750. https://doi.org/10.1109/ICDCS51616.2021.00076.
15. Sun, G.; Liao, D.; Anand, V.; Zhao, D.; Yu, H. A new technique for efficient live migration of multiple virtual machines. *Future Generation Computer Systems* **2016**, *55*, 74–86. https://doi.org/https://doi.org/10.1016/j.future.2015.09.005.
16. Rong, C.; Wang, J.H.; Wang, J.; Zhou, Y.; Zhang, J. Live Migration of Video Analytics Applications in Edge Computing. *IEEE Transactions on Mobile Computing* **2023**, pp. 1–15. https://doi.org/10.1109/TMC.2023.3246539.
17. Shi, B.; Shen, H.; Dong, B.; Zheng, Q. Memory/Disk Operation Aware Lightweight VM Live Migration. *IEEE/ACM Transactions on Networking* **2022**, *30*, 1895–1910. https://doi.org/10.1109/TNET.2022.3155935.
18. Jeong, S.; Van Tu, N.; Yoo, J.H.; Hong, J.W.K. Proactive Live Migration for Virtual Network Functions using Machine Learning. In Proceedings of the 2021 17th International Conference on Network and Service Management (CNSM), 2021, pp. 335–339. https://doi.org/10.23919/CNSM52442.2021.9615564.
19. Cho, D.; Taheri, J.; Zomaya, A.Y.; Bouvry, P. Real-Time Virtual Network Function (VNF) Migration toward Low Network Latency in Cloud Environments. In Proceedings of the 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), 2017, pp. 798–801. https://doi.org/10.1109/CLOUD.2017.118.
20. Afrasiabi, S.N.; Ebrahimzadeh, A.; Promwongsa, N.; Mouradian, C.; Li, W.; Recse, A.; Szabo, R.; Glitho, R.H. Cost-Efficient Cluster Migration of VNFs for Service Function Chain Embedding. *IEEE Transactions on Network and Service Management* **2024**, *21*, 979–993. https://doi.org/10.1109/TNSM.2023.3287757.
21. Wang, W.; Liu, Y.; Liu, J.; Li, Y.; Song, H.; Wang, Y.; Yuan, J. Consistent State Updates for Virtualized Network Function Migration. *IEEE Transactions on Services Computing* **2020**, *13*, 999–1006. https://doi.org/10.1109/TSC.2017.2765636.
22. Tang, L.; He, X.; Zhao, P.; Zhao, G.; Zhou, Y.; Chen, Q. Virtual Network Function Migration Based on Dynamic Resource Requirements Prediction. *IEEE Access* **2019**, *7*, 112348–112362. https://doi.org/10.1109/ACCESS.2019.2935014.
23. Carpio, F.; Jukan, A.; Pries, R. Balancing the migration of virtual network functions with replications in data centers. In Proceedings of the NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, 2018, pp. 1–8. https://doi.org/10.1109/NOMS.2018.8406275.
24. Das, S.; Panda, K.G.; Sen, D.; Arif, W. Maximizing Last-Minute Backup in Endangered Time-Varying Inter-Datacenter Networks. *IEEE/ACM Transactions on Networking* **2021**, *29*, 2646–2663. https://doi.org/10.1109/TNET.2021.3098766.
25. Wang, C.; Hu, Q.; Yu, D.; Cheng, X. Online Learning for Failure-Aware Edge Backup of Service Function Chains With the Minimum Latency. *IEEE/ACM Transactions on Networking* **2023**, pp. 1–15. https://doi.org/10.1109/TNET.2023.3265127.
26. Shang, X.; Huang, Y.; Liu, Z.; Yang, Y. Reducing the Service Function Chain Backup Cost Over the Edge and Cloud by a Self-Adapting Scheme. *IEEE Transactions on Mobile Computing* **2022**, *21*, 2994–3008. https://doi.org/10.1109/TMC.2020.3048885.
27. Iftikhar, A.; Qureshi, K.N.; Hussain, F.B.; Shiraz, M.; Sookhak, M. A blockchain based secure authentication technique for ensuring user privacy in edge based smart city networks. *Journal of Network and Computer Applications* **2025**, *233*, 104052. https://doi.org/https://doi.org/10.1016/j.jnca.2024.104052.

28. Duytam Ly, L.; Sadeghi Ghahroudi, M.; Ponce, V. A Systematic Literature Review of Reliable Provisioning for Virtual Network Function Chaining. *Applied Sciences* **2023**, *13*. https://doi.org/10.3390/app13095504.
29. Yang, B.; Xu, Z.; Chai, W.K.; Liang, W.; Tuncer, D.; Galis, A.; Pavlou, G. Algorithms for Fault-Tolerant Placement of Stateful Virtualized Network Functions. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–7. https://doi.org/10.1109/ICC.2018.8422444.
30. Aidi, S.; Zhani, M.F.; Elkhatib, Y. On Optimizing Backup Sharing Through Efficient VNF Migration. In Proceedings of the 2019 IEEE Conference on Network Softwarization (NetSoft), 2019, pp. 60–65. https://doi.org/10.1109/NETSOFT.2019.8806678.
31. Ishigaki, G.; Devic, S.; Gour, R.; Jue, J.P. DeepPR: Progressive Recovery for Interdependent VNFs With Deep Reinforcement Learning. *IEEE Journal on Selected Areas in Communications* **2020**, *38*, 2386–2399. https://doi.org/10.1109/JSAC.2020.3000402.
32. Qu, H.; Wang, K.; Zhao, J. Reliable Service Function Chain Deployment Method Based on Deep Reinforcement Learning. *Sensors* **2021**, *21*. https://doi.org/10.3390/s21082733.
33. Mao, W.; Wang, L.; Zhao, J.; Xu, Y. Online Fault-tolerant VNF Chain Placement: A Deep Reinforcement Learning Approach. In Proceedings of the 2020 IFIP Networking Conference (Networking), 2020, pp. 163–171.
34. Ros, S.; Tam, P.; Song, I.; Kang, S.; Kim, S. Handling Efficient VNF Placement with Graph-Based Reinforcement Learning for SFC Fault Tolerance. *Electronics* **2024**, *13*. https://doi.org/10.3390/electronics13132552.
35. Wang, W.; Tang, L.; Liu, T.; He, X.; Liang, C.; Chen, Q. Toward Reliability-Enhanced, Delay-Guaranteed Dynamic Network Slicing: A Multiagent DQN Approach With an Action Space Reduction Strategy. *IEEE Internet of Things Journal* **2024**, *11*, 9282–9297. https://doi.org/10.1109/JIOT.2023.3323817.
36. Goemans, M.X.; Williamson, D.P. The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems* **1997**, pp. 144–191.