Computer Networks 271 (2025) 111641

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Reliability-aware hybrid SFC backup and deployment in edge computing

Yue Zeng [a,b], Pan Li [c], Shanshan Lin [a], Bin Tang [d], Xiaoliang Wang [b], Zhihao Qu [d], Baoliu Ye [b,*], Song Guo [e], Junlong Zhou [a]

[a] Department of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China
[b] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[c] College of Electronic and Information Engineering, Southwest University, Chongqing, 400715, China
[d] School of Computer and Information, Hohai University, Nanjing 211100, China
[e] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China

ARTICLE INFO

Keywords:
Network function virtualization
Edge computing
5G
Approximation algorithm

ABSTRACT

As key enabling technologies for 5G, network function virtualization (NFV) abstracts services into software-based network function chains called service function chains (SFCs), greatly simplifying service management, while edge computing pushes compute resources to the edge close to IoT users, enabling low-latency services. For mission-critical applications, backup is an effective way to enhance the reliability of deployed SFCs. However, existing backup and deployment schemes mainly focus on off-site backups, neglecting on-site backups and resulting in suboptimal solutions. This paper investigates the problem of reliable SFC hybrid backup and deployment, aiming to minimize resource costs while accounting for limited edge resources and the heterogeneity of software reliability, hardware reliability, and resource charging. To tackle this problem, we first establish the mathematical association between backup and deployment decisions and these factors, formalize it as an integer nonlinear programming problem, and analyze its complexity. Then, we devise a bi-criteria approximation algorithm with rigorous theoretical guarantees, which relaxes the formalized problem to a convex optimization and rounds the fractional solution obtained by solving this convex optimization based on our insight, which approaches the optimal solution with bounded resource capacity violation, and is suitable for scenarios where moderate resource over-allocation is allowed. For cases prohibiting over-allocation, we propose a priority-guided algorithm with rigorous theoretical guarantees based on our insights, which prioritizes deploying backups for VNFs with the lowest reliability on edge sites that bring higher reliability improvements and lower charges. Extensive evaluation results show that our algorithm can save costs by up to 60.3% compared with state-of-the-art solutions.

1. Introduction

5G is envisioned to support massive emerging services, facilitated by two key technologies, edge computing and network function virtualization (NFV). Among them, edge computing brings computational resources closer to users, enabling service deployment on edge sites (ESs) for low-latency and cost-effective delivery [1–4]. Meanwhile, NFV virtualizes hardware-based network functions into software, allowing virtualized network functions (VNFs) to be flexibly deployed on commercial servers, facilitating rapid service deployment and automation [5–7]. The service can be customized by specifying a sequence of VNFs, known as a service function chain (SFC).

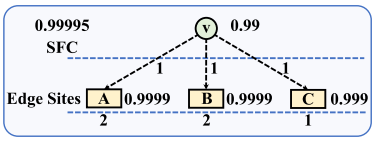
In 5G, there are many mission-critical services that demand high reliability. According to the 3GPP report [8], immersive multi-modal navigation requires 99.999% reliability,[1] while AI inference call for 99.9999% reliability. However, services deployed in ESs may be vulnerable to failure for several reasons. First, the ES hardware may be prone to failure due to low-end devices, poor operating environments, and insufficient maintenance. Second, software-based VNFs are more prone to failure than hardware-based dedicated devices [5,9]. Finally, since each SFC consists of multiple VNFs, the failure of a single VNF can disrupt the entire service. Frequent service failures may result in reduced revenue, lost potential customers, and reputational damage.

An effective way to improve SFC reliability is to deploy backups for its VNFs [10]. When a primary VNF instance fails, traffic is redirected to its backup, ensuring service continuity and masking failures.

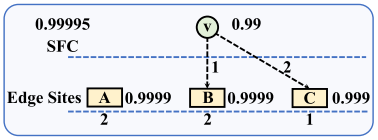
* Corresponding author.
E-mail addresses: zengyue@njust.edu.cn (Y. Zeng), lipan942907022@gmail.com (P. Li), sslin0333@gmail.com (S. Lin), cstb@hhu.edu.cn (B. Tang), waxili@nju.edu.cn (X. Wang), quzhihao@hhu.edu.cn (Z. Qu), yebl@nju.edu.cn (B. Ye), songguo@cse.ust.hk (S. Guo), jlzhou@njust.edu.cn (J. Zhou).

[1] The reliability referred to in this paper is sometimes called availability or resilience in existing research, but they are the same value for services.

https://doi.org/10.1016/j.comnet.2025.111641
Received 27 March 2025; Received in revised form 27 July 2025; Accepted 12 August 2025
Available online 20 August 2025



(a) Off-site Backup [15]



(b) Hybrid Backup

Fig. 1. An example to illustrate our motivation.

While increasing the number of backups improves reliability, it also raises resource consumption. Then, several works [11–14] investigated how many backups are needed and where VNFs should be deployed to satisfy service reliability while minimizing resource consumption. However, they only backed up the VNF instances with off-site backups, ignoring the important on-site backups, which resulted in reduced solution space and sub-optimal solutions.

For example, as shown in the motivation example in Fig. 1, there is an SFC S that requires a reliability of 0.99995, which only contains one VNF v and has a software reliability of 0.99, consuming one unit of resources (e.g., CPU and memory). There are three Edge Sites (ESs) denoted A, B, and C, are available for deployment, with hardware reliabilities of 0.9999, 0.9999, and 0.999, respectively, and resource costs of 2, 2, and 1 per unit. This SFC may fail due to VNF instance software failure or ES hardware failure. The existing off-site backup scheme [15] deploys no more than one backup on each ES, as shown in Fig. 1(a), to meet the reliability requirement of the service. This scheme deploys one instance on each of the ESs A, B, and C to satisfy the reliability of the SFC. This SFC fails only when the instances deployed on all three ESs fail. Thus, the reliability of this service can be calculated as $1-(1-0.99*0.9999)*(1-0.99*0.9999)*(1-0.99*0.999)\approx 0.9999989$, which satisfies the service reliability (without any backup instance, it cannot be done), and the corresponding resource cost is 1*2+1*2+1=5 units. In contrast, hybrid Backup allows both on-site and off-site backups, as shown in Fig. 1(b), which deploys 1 and 2 instances on ES B and C, respectively, to meet the reliability of the SFC. In this case, the SFC fails only when the instances on both ESs B and C fail. Thus, the reliability of this service can be calculated as $1-(1-0.99*0.9999)*(1-(1-(1-0.99)^2)*0.999)\approx 0.99998889$, which satisfies the service reliability (without any backup instance, it also cannot be done), and the corresponding resource cost is 1*2+2*1=4 units. Obviously, hybrid backup offers a cost-effective way to ensure service reliability by leveraging both on-site and off-site backups.[2]

This paper investigates how to hybrid back up and deploy reliable SFC cost-effectively, considering intrinsic and critical factors, including hardware and software reliability and charging heterogeneity. The challenge lies in the coupling between backup and deployment decisions, which jointly determine service reliability and resource costs. What is more, ESs are heterogeneous in hardware reliability and charging, and VNFs in SFC are heterogeneous in resource consumption and software reliability. To capture all these characteristics, we establish a mathematical relationship between hybrid backup and deployment decisions and these factors, formulating the problem as an integer nonlinear program and rigorously proving that determining its feasibility is strongly NP-hard. As a compromise, we develop a bi-criteria approximation algorithm based on convex relaxation and rounding, which guarantees an approximation ratio for resource cost while allowing only bounded capacity constraint violations. This algorithm first relaxes the problem as a convex problem by relaxing its integer variable constraints, and then rounds the fractional solution based on our insights to achieve a resource-efficient solution. This solution appropriately over-allocates resources and is reasonable and widely adopted to improve resource utilization, but may incur performance penalties for sustained high loads. To handle scenarios where resource over-allocation is prohibited, we further propose a priority guidance algorithm, which is designed based on our insight to prioritize VNFs with lower reliability to deploy instances on ESs that bring higher reliability improvement and lower charges. Finally, we verified the superiority and optimality of our algorithm in terms of resource cost by extensive evaluation results.

The main contributions of this paper are as follows:

- To the best of our knowledge, we are the first to study how to hybrid back up and deploy reliable SFC cost-effectively, which considers intrinsic and critical factors, including hardware/software reliability and resource charging heterogeneity. To capture all these characteristics, we establish a mathematical connection between the hybrid backup and deployment decision and these factors, formalize it as an integer nonlinear programming, and analyze its complexity.
- To tackle this problem, we first propose a bi-criteria approximation algorithm to approximate the optimal objective with a proven approximation ratio and bounded capacity violation, suitable for scenarios where moderate resource over-allocation is allowed. To handle scenarios where resource over-allocation is prohibited, we devise a priority-guided algorithm with rigorous theoretical guarantees based on our insights, which prioritizes deploying backups for VNFs with the lowest reliability on ESs that bring higher reliability improvements and lower charges.
- Extensive evaluation results show that, when compared with the state-of-the-art solutions, our proposed algorithms can reduce costs by up to 60.3%.

The structure of the paper is as follows. Section 2 provides a brief overview of the related work. The studied problem is formulated in Section 3. Sections 4 and 5 present the bi-criteria approximation algorithm and the priority guidance algorithm, respectively, which are then evaluated in Section 6. Finally, Section 7 concludes the paper.

2. Related work

This section elaborates on related works and distinguishes them from this paper as follows.

2.1. Reliable service deployment

The concept of NFV was first defined in [27], which separates network functions from dedicated hardware devices and greatly facilitates service deployment. Then, a series of works [16–18] studies how to deploy services cost-efficiently. As pioneers, Cohen et al. [16] studied how to deploy VNFs to minimize resource costs and proposed a linear relaxation-based bi-criteria approximation algorithm. Considering reliability requirements, Yala et al. [18] further studied how to trade off between service reliability and delay performance and proposed an efficient heuristic algorithm. Moreover, Martin et al. [17] studied service deployment and data routing to meet service reliability while saving resource costs. These studies greatly help service providers to supply reliable services in a cost-efficient manner. However, they only deploy the primary VNF without backup, leading to infeasible solutions for services with long SFC and high-reliability requirements.

[2] It should be noted that too many on-site backups can lead to a single point of failure, where one ES hardware fails, then all VNFs on that ES fail. Therefore, an effective combination of on-site and off-site backups is critical.

Table 1
Summary of related research.

Related work	SFC	Deployment	Backup	Onsite & Offsite	Software & Hardware	No training
[16]	✗	✓	✗	✗	✗	✓
[17]	✓	✓	✗	✗	✗	✓
[18]	✓	✓	✗	✗	✓	✓
[11–14,19–22]	✓	✓	✓	✗	✗	✓
[23]	✓	✓	✓	✓	✗	✓
[24,25]	✗	✓	✓	✓	✓	✓
[15,26]	✓	✓	✓	✗	✓	✗
This work	✓	✓	✓	✓	✓	✓

2.2. Reliable service backup and deployment

As pioneers, Potharaju et al. [19] investigated the failure of network functions and verified that a static 1:1 backup scheme could effectively improve service reliability. For high-reliability performance, Kanizo et al. [20] proposed a novel static backup and dynamic deployment scheme to maximize the expected service reliability. For resource efficiency, several research efforts [11–13] have investigated how to deploy and back up VNFs to meet service reliability while saving resources. Fan et al. [11,12] studied how to dynamically deploy SFC with on-site backup [12] or off-site backup [12] to meet service heterogeneous reliability requirements while minimizing the number of backups and proposed efficient online heuristic algorithms. Considering the heterogeneity of VNFs in resource consumption, Zhang et al. [13] further studied the backup and deployment of SFC to minimize the required backup resources. Besides, Peng et al. [21] and Zheng et al. [22] studied how to backup the VNF components of SFC, for reliable SFC. Moreover, Zheng et al. [23] further investigated the deployment of reliable services through shared protection. These efforts have greatly facilitated reliable services in highly reliable data centers. However, they ignore hardware reliability, which fails to meet the demands of mission-critical services requiring high reliability in low-reliability edge environments.

There are also several studies [15,24,26] that focus on strategies for backing up and deploying reliable services in edge environments. Considering both hardware and software reliability, new hybrid service backup and deployment algorithms [24,25] are proposed to protect the reliability of VNFs while ensuring resource efficiency. However, these solutions are aimed at services with a single VNF and fail to support the widely used complex services (SFC) with multiple VNFs. Furthermore, Zeng et al. [15,26] studied the reliable SFC provisioning problem considering the complex SFC structure and heterogeneous hardware resource charges and designed an off-site backup mechanism based on deep reinforcement learning (DRL). However, they only backed up the VNF instances with off-site backups, ignoring the important on-site backups, which resulted in reduced solution space and sub-optimal solutions. Besides, their DRL-based solutions introduce high-cost retraining (e.g., thousands of rounds of offline training) to adapt to various scenarios.

Different from the above studies, this paper explores the hybrid backup and deployment of SFCs to meet the required reliability while minimizing resource costs. It takes into account critical factors such as hardware and software reliability and charging heterogeneity. To highlight the novelty of this paper, we summarize related research as shown in Table 1.

3. System model and problem formulation

This section introduces the system model, formalizes the studied problem, and analyzes its complexity. The notations to be used are provided in Table 2.

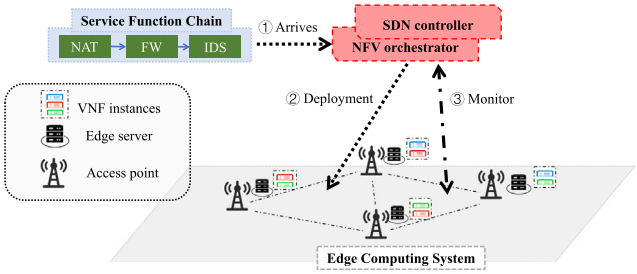


Fig. 2. NFV-enabled edge computing system.

Table 2
Summary of notations.

Notations	Descriptions
$\mathbb{E}$	Set of ESs
$\mathbb{V}$	Set of VNFs required by $\mathcal{S}$
r_e	Hardware reliability of ES e
r_v	Software reliability of VNF v
K_e	Computing resource capacity on ES e
κ_v	Computing resources required by VNF v
τ_e	Cost of unit computing resources on ES e
$\mathcal{R}$	Reliability required by SFC $\mathcal{S}$
R	Reliability of SFC $\mathcal{S}$
R_v	Reliability of VNF v
$R_{v,e}$	Reliability of VNF v on ES e
δ	Maximum number of allowed instances (primary or backup) for component VNF
Decisions	**Descriptions**
$n_{v,e}$	Integer variable indicates the number of VNF instances deployed for VNF v on ES e

3.1. System model

With reference to the ESI-defined NFV architecture [28], our NFV-enabled edge computing system (NFV-ECS), illustrated in Fig. 2, is composed of the following three components.

- **Service.** Each service is generated by IoT devices and can be abstracted into an SFC, which arrives dynamically and has specific reliability requirements. Once an SFC request arrives, it needs to be deployed and backed up to ESs. Since each SFC may consist of multiple VNFs, it can be abstracted as a two-tuple $\mathcal{S} = \langle \mathbb{V}, \mathcal{R} \rangle$, where $\mathbb{V}$ denotes the set of its component VNFs, and $\mathcal{R}$ denotes its reliability requirements. Besides, each VNF $v \in \mathbb{V}$ has two attributes, software reliability, and computing resource requirements, denoted as r_v and κ_v, respectively.[3]

[3] Similar to existing work [24,29,30], we consider computing resources as the bottleneck. Because VNFs are deployed in lightweight containers, they need to support line-speed processing and forwarding, where computing resources are scarce.

- **MANO.** The Service Management and Orchestration (MANO), owned by the service provider, includes the NFV orchestrator and the SDN controller. The former is responsible for backing up and deploying reliable services, reliably monitoring them, and initiating failure recovery after failure is detected. The latter is responsible for rerouting traffic when network devices or VNFs fail. As the MANO is provided by fault-tolerant SDN controller [31–33], it is considered to be reliable. After a service is deployed, MANO does not involve normal operations to avoid becoming a performance bottleneck [10].
- **NFVI.** In the NFV-Edge Cloud System (NFV-ECS), the NFV Infrastructure (NFVI) consists of a set of Edge Servers (ESs) $\mathbb{E}$. For each ES $e \in \mathbb{E}$, its capacity is denoted as K_e, its reliability is denoted as r_e, and it charges τ_e per unit of resource.

3.2. Problem definition

In this study, we explore how to back up and deploy VNFs in an SFC within resource-constrained edge environments, aiming to meet reliability requirements while minimizing resource costs. This problem is referred to as the Reliable SFC Backup and Deployment (RSBD) problem, which can be formally defined as follows.

Definition 1. Given a set $\mathcal{E}$ of ESs and an SFC request generated by IoT devices, where each ES e has resource capacity K_e and hardware reliability r_e. The SFC request has a reliability requirement $\mathcal{R}$ and consists of a set $\mathbb{V}$ of VNFs. Each VNF $v \in \mathbb{V}$ has software reliability r_v and can be deployed with no more than δ instances as primary or backup instances. Moreover, each ES $e \in \mathbb{E}$ charges τ_e for each unit of resource. The RSBD problem aims to determine the optimal number of primary and backup instances for each VNF to deploy at the appropriate ESs such that the reliability requirement of the SFC is met, while minimizing the total resource costs.

It should be noted that in our research, we mainly focus on applications with stringent reliability requirements but relatively lenient delay constraints, where the inter-edge-site latency can be disregarded in our modeling.[4]

3.3. Problem formulation

RSBD problem aims to cost-effectively deploy and backup the component VNFs of SFC in resource-constrained ESs to meet their reliability constraints. Next, we formalize the cost model, reliability constraints, and capacity constraints, as follows.

Cost Model. Referring to [13],[5] the total resource cost involves the cost incurred by deploying the primary or backup instances of all component VNFs on all ESs, which can be modeled as

$$\sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}}\tau_e\kappa_v n_{v,e},$$

where $n_{v,e}$ denotes the number of primary or backup instances deployed for VNF v on ES e.

Reliability Constraints: Reliability can be quantified using the Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR) [11,13,39], with the reliability formula given as $r = \frac{MTBF}{MTBF+MTTR}$. Obviously, the reliability value is between 0 and 1, where a higher value indicates greater reliability and a lower likelihood of failure.

Each SFC may contain multiple VNFs, and if any VNF fails, the entire SFC is disrupted [15]. The virtual link between VNFs is assumed to be reliable [12,13,15,24,26].[6] Then, similar to [12,13,15,24,26], the reliability of SFC can be described as

$$R = \prod_{v\in\mathbb{V}} R_v, \tag{1}$$

where R_v indicates the reliability of VNF v.

To achieve high reliability, the instance of each VNF can be deployed across multiple ESs. In this scenario, a VNF will only fail when it fails on all ESs simultaneously. Therefore, the reliability of VNF v can be described as

$$R_v = 1 - \prod_{e\in\mathbb{E}}(1 - R_{v,e}), \forall v \in \mathbb{V}, \tag{2}$$

where $R_{v,e}$ indicates the reliability of VNF v on ES e.

A VNF on an ES fails only if the ES fails or all instances of the VNF on that ES fail. Therefore, the reliability of VNF v on ES e can be denoted as

$$R_{v,e} = r_e(1 - (1 - r_v)^{n_{v,e}}), \forall v \in \mathbb{V}, e \in \mathbb{E}. \tag{3}$$

The reliability of SFC needs to meet its reliability expectations. Then, based on Eq. (1), we have

$$\prod_{v\in\mathbb{V}} R_v \geq \mathcal{R}. \tag{4}$$

SFC Completeness Constraints: To ensure the completeness of SFC, we need to deploy at least one instance for each VNF $v \in \mathbb{V}$ as its primary instance, which can be expressed as

$$\sum_{e\in\mathbb{E}} n_{v,e} \geq 1, \forall v \in \mathbb{V}. \tag{5}$$

Capacity Constraints: The VNF instance deployed on each ES should not exceed its capacity. This constraint can be expressed as

$$\sum_{v\in\mathbb{V}} \kappa_v n_{v,e} \leq K_e, \forall e \in \mathbb{E}. \tag{6}$$

Hybrid Backup Constraints: Hybrid backup means that both on-site and off-site backup are allowed, which means that each ES v can provide multiple instances of each VNF v. To control excessive resource consumption, we limit the number of primary or backup instances of each component VNF to δ. This constraint can be expressed as

$$\sum_{v\in\mathbb{V}} n_{v,e} \leq \delta, \forall e \in \mathbb{E}, \tag{7}$$

[4] For instance, as reported by 3GPP [8], immersive multi-modal navigation applications typically require a reliability of 0.99999 and can tolerate up to 400 ms of latency, while AI inference services demand a reliability of 0.9999 with latency thresholds around 1 s. ESs (small edge clouds) are typically deployed a few to tens of kilometers apart [34]. As modeled by Zhang et al. [35], the average latency between ESs can be approximated by the equation $\mathcal{L} = 0.019x_{dist} + 23.3$, x_{dist}, where x_{dist} denotes the distance between two ESs. Based on this model, the inter-edge-site latency is less than $0.019 * 100 + 23.3 = 25.2$ ms. Additionally, the processing delay of a typical VNF is less than 1 ms [36]. Since a typical SFC consists of 1 to 7 VNFs [37], the worst-case scenario involves 6 inter-edge-site transmissions and 7 VNF executions, resulting in a total latency of $6 * 25.2 + 7 * 1 = 158.2$ ms. This is still well within the end-to-end latency budgets (400–1000 ms) of the target applications and thus does not need to be explicitly modeled as a constraint in our problem formulation.

[5] This cost model is reasonable because in commercial serverless computing platforms such as AWS Lambda, an instance is charged based on its memory resource configuration, and computing resources are allocated proportionally, and vice versa.[38]

[6] similar to [12,13,15,24,26], we focus on how to enhance the reliability of the SFC composed of multiple VNFs, whereas the virtual links connecting them are considered to be reliable and the routing between them is outside of our considerations, which is reasonable for the following reasons. The virtual link connecting VNFs is implemented by the underlying carrier-class network, which guarantees 99.999% or 99.9999% reliability [40]. Furthermore, the routing of these virtual links can be achieved using mature software-defined networking (SDN)-based multipath routing algorithms [41], which can further enhance the reliability of these virtual links. Moreover, once these virtual links fail, they can be quickly restored within about 10 ms by replanning the underlying routing. Therefore, it is reasonable to consider virtual links as reliable. Besides, the routing between VNFs can be achieved through existing mature multi-path routing algorithms [41], which are beyond our consideration for simplicity.

$$n_{v,e} \in \{0, 1, \dots, \delta\}, \forall v \in \mathbb{V}, e \in \mathbb{E}. \tag{8}$$

Thus, the RSBD problem can be formulated as

$$\min_{n_{v,e}} \sum_{v\in\mathbb{V}} \sum_{e\in\mathbb{E}} \tau_e \kappa_v n_{v,e}$$

s.t. (2), (3), (4),(5), (6), (7), (8).

3.4. Hardness result

Theorem 1. *Determining whether the RSBD problem has a feasible solution is strongly NP-hard.*

Proof. This theorem can be proved by demonstrating that a special case of the RSBD problem is equivalent to the minimization version of the generalized assignment problem (MinGAP) [42], which is known to be NP-hard. We assume that the reliability requirement of each SFC is 0, that is, $R = 0$. Besides, the maximum number of instances allowed for each VNF is 1, that is, $\delta = 1$. In this case, the RSBD problem is equivalent to the MinGAP problem. Since determining the feasibility of MinGAP has been proven to be strongly NP-hard [42], it follows that establishing the feasibility of the RSBD problem is also strongly NP-hard. □

The above theorem proves that our problem is strongly NP-hard by showing that a special case of our problem is equivalent to a classical strongly NP-hard problem, such as the MinGAP [42]. This suggests that no polynomial-time algorithms exist for the RSBD problem, unless $P = NP$.

Fortunately, there are existing bi-criteria approximation algorithms [43] that can effectively solve the MinGAP problem, which approach the optimal solution of the MinGAP problem with moderate resource capacity violations. This inspires us to design a similar bi-criteria approximation algorithm for the RSBP problem. However, compared to the MinGAP problem, the RSBD problem is more challenging due to more constraints, especially nonlinear reliability constraints. This inspires us to design a bi-criteria approximation algorithm tailored for the RSBD problem in the next section.

4. Bi-criteria approximation algorithm

This section first simplifies the RSBD problem by relaxing it to a convex optimization problem, then designs a bi-criterion approximation algorithm based on relaxation and rounding, and finally proves its theoretical properties.

4.1. Algorithm design

We first relax the RSBD problem from an integer optimization to a convex optimization by relaxing its integer variables. Then, we solve this convex optimization to inspire backup and deployment decisions.

4.1.1. Problem relaxation

First, we relax the integer constraint Eq. (8) to

$$0 \le n_{v,e} \le \delta, \forall v \in \mathbb{V}, e \in \mathbb{E}. \tag{9}$$

Then, the relaxed RSBD (R-RSBD) problem can be formulated as follows

$$\min_{n_{v,e}} \sum_{v\in\mathbb{V}} \sum_{e\in\mathbb{E}} \tau_e \kappa_v n_{v,e}$$

s.t. (2),(3), (4), (5), (6), (7), (9).

Theorem 2. *R-RSBD problem is convex.*

Proof. See Appendix A. □

4.1.2. Rounding

After solving R-RSBD, a fractional solution can be obtained. Then, we need to round this fractional solution to an integer solution that ensures reliability while minimizing resource cost. This involves deciding which VNF should have a backup and on which ES it should be deployed. We offer the following insights to guide the rounding process.

Proposition 1. *The reliability of the SFC is determined by the VNF with the lowest reliability.*

Proof. As shown in Eq. (1), the reliability of the SFC is the product of the reliabilities of the individual VNFs, where the reliability of each VNF $R_v \in (0, 1)$. Therefore, the overall reliability $R = \prod_{v\in\mathbb{V}} R_v$ is always less than or equal to the minimum reliability of any individual VNF, i.e., $R \le \min R_v, \forall v \in \mathbb{V}$. This means the SFC reliability is constrained by the VNF with the lowest reliability. □

Based on this proposition, we are motivated to add backup instances for the VNF with the lowest reliability. In other words, we round up the corresponding variable for the VNF with the lowest reliability.

Next, we need to determine which ES to use to deploy the backup instances for the selected VNF. In the fractional solution, a larger value means more preference for the optimal solution. Therefore, we deploy the newly added VNF instance to that ES with the larger value, i.e., round the variable.

4.1.3. Algorithm description

The detailed process of relaxation rounding is outlined in **Algorithm 1**. Initially, we initialized the backup and deployment decision and initialized the SFC service as rejected. Then, we try to solve the R-RSBD problem and obtain its optimal solution. Once there is no feasible solution, we reject the SFC request. Otherwise, the request is accepted, and based on the aforementioned insight, we greedily deploy instances on the ES with the highest value for the VNF with the lowest reliability, continuing until the reliability requirements for the VNF are satisfied.

Algorithm 1 Relaxation Rounding Algorithm

Input: Parameters in R-RSBD.
Output: $\mathcal{A}$: boolean variable indicating whether the SFC request is accepted; $n_{v,e}$: VNF backup and deployment decision.

```
Initialize n_{v,e} ← 0, ∀e ∈ 𝔼, v ∈ 𝕍;
𝒜 ← False;
R ← 0; R_v ← 0, ∀v ∈ 𝕍;
Solve R-RSBD and get its optimal solution ñ_{v,e};
n_{v,e} ← ⌊ñ_{v,e}⌋, ∀e ∈ 𝔼, v ∈ 𝕍;
if R-RSBD is feasible then
    𝒜 ← True;
    Ψ ← {(v, e), ∀e ∈ 𝔼, v ∈ 𝕍};
    while R ≤ ℛ do
        v ← arg min_{v∈𝕍} R_v;
        e ← arg max_{(v,e)∈Ψ} ñ_{v,e};
        n_{v,e} ← ⌈ñ_{v,e}⌉;
        Ψ ← Ψ − (v, e);
        Update R and R_v using Eqs. (1), (2), (3);
    end while
end if
```

The structure of the SFC backup and deployment (SBD) algorithm is outlined in **Algorithm 2**. Once an SFC request arrives, the algorithm is triggered. The algorithm first calls **Algorithm 1** to obtain the backup and deployment decision. Once R-RSBD is infeasible, the SFC request is rejected; otherwise, the SFC request is accepted, and the corresponding deployment and backup decisions are implemented. Finally, the available resources are updated.

Algorithm 2 SFC Backup and Deployment Algorithm

Input: Parameters in RSBD.
Output: $\mathcal{A}$: boolean variable indicating whether the SFC request is accepted; $n_{v,e}$: VNF backup and deployment decision.

```
while A new SFC request S arrives do
    A, n_{v,e} ← Algorithm 1;
    if A = True then
        Accept the SFC request S, and deploy and backup it accordingly;
        K_e ← K_e − Σ_{v∈V} k_v n_{v,e}, ∀e ∈ E;
    else
        Reject the SFC request S;
    end if
end while
```

4.2. Algorithm analysis

Theorem 3. *The SBD algorithm runs in polynomial time.*

Proof. First, we examine **Algorithm 1**, which requires $O(|\mathbb{V}\parallel\mathbb{E}|)$ to initialize its parameters. The algorithm then proceeds to solve the R-RSBD problem. As demonstrated in Theorem 2, the R-RSBD problem is convex, and it can be efficiently solved in polynomial time using the interior point method [44]. If the R-RSBD problem is found to be infeasible, the algorithm terminates. Otherwise, it performs rounding on the fractional solution to convert it into an integer solution. Specifically, the algorithm consumes $O(|\mathbb{V}\parallel\mathbb{E}|)$ to initialize the backup decision as the optimal solution that is rounded down and initialize the rounded candidate set. Then, the algorithm iteratively rounds up the elements in the candidate set. Since the R-RSBD problem is feasible, rounding up the elements in the candidate set can obviously satisfy the reliability constraint and terminate. In the worst case, it iterates $O(|\mathbb{V}\parallel\mathbb{E}|)$ rounds, each iteration consumes $O(|\mathbb{V}\parallel\mathbb{E}|)$. Therefore, Algorithm 1 runs in polynomial time. Since **Algorithm 1** runs in polynomial time, SBD algorithm (**Algorithm 2**) obviously runs in polynomial time. □

Before analyzing the approximation performance of the bi-criteria approximation algorithm, we formally define it as follows [45].

Definition 2 (*Bi-Criteria Approximation*).. An algorithm achieves (ρ,ϕ)-approximation to RSBD if the following two conditions are satisfied simultaneously.

- It requires at most ρ times the minimum resource cost.
- It violates the capacity constraint no more than ϕ times.

Theorem 4. *SBD algorithm is a* $(\lfloor\log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}})+1\rfloor\frac{\tau_{max}}{\tau_{min}}, 1+\frac{\sum_{v\in\mathbb{V}}\kappa_v}{K_e})$ *bi-criteria approximation algorithm, where* $\tau_{max}=\max_{v\in\mathbb{V}}\{\tau_v\}$, $\tau_{min}=\min_{v\in\mathbb{V}}\{\tau_v\}$, $\xi=\min_{v\in\mathbb{V}}\{r_v\}$, $\eta=\min_{e\in\mathbb{E}}\{r_e\}$.[7]

Proof. See Appendix B. □

The above SBD algorithm shows good approximation properties with moderate capacity violations. A moderate capacity violation implies a moderate over-allocation of resources, which is reasonable and widely adopted to improve resource utilization. In our RSBD problem, the backup instance is activated only when the primary instance fails, so appropriate over-allocating resources help improve resource utilization. We further validated that our SBD algorithm can approach the optimal solution well while only slightly violating the constraints in Figs. 4 and 7. However, over-allocating resources may cause performance degradation, which is not suitable for scenarios where services have stringent performance requirements and resource over-allocation is prohibited. Therefore, we next design a heuristic algorithm that explores a solution that guarantees all constraints.

5. Priority-guided algorithm

This section first presents insights obtained based on formulas, then designs a Priority-Guided (PG) algorithm based on it, and finally analyzes its theoretical properties.

5.1. Algorithm design

We first provide the insight for algorithm design and then introduce the algorithm in detail.

5.1.1. Insight

Our RSBD problem aims to minimize resource costs while satisfying reliability constraints, capacity constraints, cost constraints, decision variable constraints, and SFC completeness constraints. The core lies in how to satisfy reliability constraints while saving resources. Although deploying more backups brings higher reliability, it also comes with higher resource costs. Therefore, a key challenge is identifying which VNF to back up and on which ES. The insights presented in Proposition 1 guided us to prioritize adding instances for the VNF with lower reliability.

Next, we need to consider which ES should the instance of this VNF be deployed to. Referring to [15], we can define the priority of each ES as

$$I_e=\frac{\overrightarrow{R}_e-R}{\tau_e},\forall e\in\mathbb{E}, \tag{10}$$

where $\overrightarrow{R}_e$ denotes the SFC reliability after adding an instance for VNF v on ES e. The motivation behind this formula is to prioritize deploying functional instances to ESs that bring higher reliability improvements and lower resource costs.

We further refine the definition of ES priority I_e based on two key insights. First, if deploying a functional instance on any of the available ESs can satisfy the SFC reliability requirement, it is preferable to select the site with lower resource cost, as improving reliability becomes less critical in this case. Second, when not all primary instances of the component VNFs in the SFC are deployed, the overall SFC reliability drops to zero. In such scenarios, the metric $I_e=0$ fails to capture the relative value of different ESs. Therefore, it is more appropriate to prioritize ess with lower resource costs. Based on these observations, we revise the ES priority definition to

$$I_e=\begin{cases}\frac{\overrightarrow{R}_e-R}{\tau_e},\forall e\in\mathbb{E}, & \text{if } \overrightarrow{R}_e<\mathcal{R} \text{ and } \overrightarrow{R}_e>0,\\ \frac{\Delta}{\tau_e}, & \text{otherwise,}\end{cases} \tag{11}$$

where Δ is a positive integer constant, $\Delta\gg 1$.

Based on the above insights and designs, we can avoid frequent deployment of backups for highly reliable VNFs, and backups will not always be deployed to highly reliable ESs, but instead lead to greater reliability improvements and low-cost ESs.[8]

5.1.2. Algorithm description

[7] In this theorem, we define $\tau_{max}=\max_{v\in\mathbb{V}}\{\tau_v\}$, $\tau_{min}=\min_{v\in\mathbb{V}}\{\tau_v\}$, $\xi=\min_{v\in\mathbb{V}}\{r_v\}$, $\eta=\min_{e\in\mathbb{E}}\{r_e\}$, which are defined variables, not assumptions.

[8] For example, consider an SFC composed of two VNFs, $VNF1$ and $VNF2$, both with a software reliability of 0.99. There are three ESs, A, B, and C, with reliabilities of 0.9999999, 0.99, and 0.999, respectively. Besides, $VNF1$ and $VNF2$ have each been deployed on ESs A and B, respectively. $VNF1$, and $VNF2$ have already deployed an instance on ESs A and B, respectively. Obviously, $VNF2$ has lower reliability, it is the bottleneck of SFC reliability and it is more worthwhile to deploy backups for it.

Algorithm 3 Priority-Guided Algorithm

Input: Parameters in RSBD.
Output: $\mathcal{A}$: boolean variable indicating whether the SFC request is accepted; $n_{v,e}$: VNF backup and deployment decision.

```
1: while A new SFC request S arrives do
2:   Initialize R ← 0; n_{v,e} ← 0, ∀e ∈ 𝔼, v ∈ 𝕍;
3:   while R < ℛ do
4:     Calculate R_v by Eq. (2) and Eq. (3);
5:     v ← arg min{R_v, ∀v};
6:     ℂ ← {e, ∀e ∈ 𝔼 | K_e ≥ k_v, n_{v,e} < δ};
7:     if ℂ! = ∅ then
8:       Calculate I_e, ∀e ∈ ℂ by Eq. (11);
9:       e ← arg max{I_e, ∀e ∈ ℂ};
10:      n_{v,e} ← n_{v,e} + 1;
11:      Update R, K_v by Eqs. (1), (2), (3), (7);
12:    else
13:      Break;
14:    end if
15:  end while
16:  if R ≥ ℛ then
17:    𝒜 ← True; Accept the SFC request S, and deploy and backup
       it accordingly;
18:  else
19:    𝒜 ← False; Reject the SFC request S;
20:  end if
21: end while
```

The algorithm we designed is shown in **Algorithm 3**, which is triggered when a new SFC request arrives. We first initialize the SFC reliability and the VNF backup and deployment decision. Then, we calculate the reliability of each VNF and obtain the one with the lowest reliability. Further, we determine which candidate ESs have the capacity to accommodate the instance of this VNF. If the candidate ES set is not empty, then we evaluate the priority of each ES by Eq. (11) and deploy the instance of this VNF to that highest priority ES. Then, we iterate the above steps until the SFC reliability is satisfied or there are no candidate ESs for further accommodating the VNF instance with the lowest reliability. Finally, once the SFC's reliability is satisfied, we backup and deploy it accordingly. Otherwise, we reject the SFC request.

5.2. Algorithm analysis

Theorem 5. *PG algorithm runs in* $O(\delta|\mathbb{V}|^2|\mathbb{E}| + \delta|\mathbb{V} \parallel \mathbb{E}|^2)$.

Proof. The algorithm begins by initializing key parameters, which takes $O(|\mathbb{V} \parallel \mathbb{E}|)$. It then iteratively assigns backup instances to the SFC and deploys them to high-priority ESs. In the worst case, up to δ instances may be deployed per ES for each VNF, leading to at most $O(\delta|\mathbb{V} \parallel \mathbb{E}|)$ iterations. Each iteration has a computational cost of $O(|\mathbb{V}| + |\mathbb{E}|)$, resulting in an overall complexity of $O(\delta|\mathbb{V}|^2|\mathbb{E}| + \delta|\mathbb{V} \parallel \mathbb{E}|^2)$ (lines 3–15). Thus, the PG algorithm runs within this complexity bound. □

In edge computing environments, although the capacity of each ES is limited, the massive distributed ESs provide sufficient resources for the backup and deployment of the SFC component VNFs [15]. Therefore, without loss of generality, we assume that ESs have sufficient capacity to accept the SFC request. Next, we analyze the performance of the PG algorithm when the SFC request is accepted.

Theorem 6. *Assuming the SFC request is accepted, PG approximates the optimal solution with a factor* $(\lfloor \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor \frac{\tau_{max}}{\tau_{min}})$ *while guaranteeing all constraints, where* $\tau_{max} = \max_{v\in\mathbb{V}}\{\tau_v\}$, $\tau_{min} = \min_{v\in\mathbb{V}}\{\tau_v\}$, $\xi = \min_{v\in\mathbb{V}}\{r_v\}$, $\eta = \min_{e\in\mathbb{E}}\{r_e\}$.

Proof. We first prove that PG algorithm guarantees all constraints, including the reliability constraint, capacity constraint, SFC completeness constraint and hybrid backup constraint. As shown in **Algorithm 3**, our algorithm accepts an SFC request only when its reliability is satisfied. Naturally, the reliability of an accepted SFC request is satisfied. Besides, the number of instances of each VNF is initialized to 0 and is increased one by one without exceeding δ (line 6), which meets the hybrid backup constraint. As in Eqs. (1), (2), and (3), once a VNF is not deployed with any instances, its reliability is 0, and the reliability of the SFC is also 0, which violates the reliability constraint. Therefore, for an accepted SFC, the SFC completeness constraint is also satisfied. Finally, before deploying an instance on an ES for a VNF, we check whether its capacity can accommodate the VNF instance (line 6). The instance can be deployed only when the capacity constraint is not violated. Therefore, all constraints are satisfied.

Next, we discuss the approximate performance of PG algorithm for an accepted SFC as follows. As shown in PG algorithm, we greedily deploy instances for the least reliable VNF until service reliability is met. The Proposition 2 (in Appendix B) is also hold for PG algorithm. Specifically, when a VNF receives additional instances, its reliability remains below $\mathcal{R}^{\frac{1}{|\mathbb{V}|}}$. Thus, we establish the following

$$R'_v < \mathcal{R}^{\frac{1}{|\mathbb{V}|}}, \tag{12}$$

where R'_v represents the reliability of VNF v just before the addition of its final backup instance, $R'_v = 1 - \prod_{e\in\mathbb{E}}(1 - r_e(1-(1-r_v)^{n'_{v,e}}))$.

Based on Eqs. (2), (3), we have

$$\begin{aligned} 1 - R'_v &\le \prod_{e\in\mathbb{E}}(1-\eta(1-(1-\xi)^{n'_{v,e}})) \le \prod_{e\in\mathbb{E}}(1-\xi\eta)^{\frac{n'_{v,e}}{\delta}} \\ &= (1-\xi\eta)^{\frac{\sum_{e\in\mathbb{E}} n'_{v,e}}{\delta}}, \end{aligned} \tag{13}$$

where $n'_{v,e}$ denotes the backup and deployment decision for VNF v before its last instance is added. The second inequality is because when $n'_{v,e} = 0$, it clearly holds, while when $n'_{v,e} \in \{1,\dots,\delta\}$, So $1-\eta(1-(1-\xi)^{n'_{v,e}}) \le 1-\eta(1-(1-\xi)) = 1-\eta\xi \le (1-\eta\xi)^{\frac{n'_{v,e}}{\delta}}$.

Since $1 \ge 1-\xi\eta \ge 0$, by taking the logarithm of Eq. (13), we have

$$\sum_{e\in\mathbb{E}} n'_{v,e} \le \log_{1-\xi\eta}(1-R'_v) \le \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}), \tag{14}$$

where the second equality is based on Eq. (12).

As defined for $n'_{v,e}$ in Eq. (B.1), we have

$$\sum_{e\in\mathbb{E}} n_{v,e} \le \sum_{e\in\mathbb{E}} n'_{v,e} + 1 \le \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1, \forall v \in \mathbb{V}. \tag{15}$$

Since $n_{v,e}, \forall e \in \mathbb{E}$ are integers, then $\sum_{e\in\mathbb{E}} n_{v,e}$ is also an integer, so

$$\sum_{e\in\mathbb{E}} n_{v,e} \le \lfloor \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor, \forall v \in \mathbb{V}. \tag{16}$$

Based on the above analysis, the approximate ratio of PG algorithm is

$$\begin{aligned} \beta &= \frac{SOL_P}{OPT_R} = \frac{\sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_e\kappa_v n_{v,e}}{\sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_e\kappa_v n^*_{v,e}} \\ &\le \frac{\sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_{max}\kappa_v n_{v,e}}{\sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_{min}\kappa_v n^*_{v,e}} \\ &= \frac{\tau_{max}\sum_{v\in\mathbb{V}}(\kappa_v\sum_{e\in\mathbb{E}} n_{v,e})}{\tau_{min}\sum_{v\in\mathbb{V}}(\kappa_v\sum_{e\in\mathbb{E}} n^*_{v,e})} \\ &\le \frac{\tau_{max}\sum_{v\in\mathbb{V}}(\kappa_v\sum_{e\in\mathbb{E}} n_{v,e})}{\tau_{min}\sum_{v\in\mathbb{V}}\kappa_v} \\ &\le \frac{\tau_{max}\sum_{v\in\mathbb{V}}(\kappa_v\lfloor\log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1\rfloor)}{\tau_{min}\sum_{v\in\mathbb{V}}\kappa_v} \\ &\le \lfloor\log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1\rfloor\frac{\tau_{max}}{\tau_{min}}. \end{aligned} \tag{17}$$

where SOL_P indicates the solution of PG algorithm. The third equation is because $\tau_{max} = \max_{v\in\mathbb{V}}\{\tau_v\}$, $\tau_{min} = \min_{v\in\mathbb{V}}\{\tau_v\}$. The fifth and sixth inequalities are obtained by Eqs. (6), (16). □

It should be noted that we also verified that PG algorithm can approach the optimal solution well, and its results are shown in Fig. 3.

6. Evaluation

This section first introduces the evaluation settings and then discusses the evaluation results.

6.1. Evaluation settings

Evaluation Environment: We develop a Python-based simulator and run all experiments on a machine equipped with a 24-core 13th Gen Intel(R) Core(TM) i9-13900K CPU (3.0 GHz) and 32 GB RAM.

Service Function Chains. The length of each SFC is randomly chosen between 1 and 7, aligning with the typical SFC length range reported in [37]. Their reliability requirements are randomly selected in [0.999, 0.9999, 0.99999, 0.999999], which is derived from the popular service reliability requirements [8]. For each VNF, its computing resource demand is distributed in [40, 100] MHz, and its software reliability is distributed in $\mathbb{R}_v = [0.999, 0.99999]$[19,24].

Edge Sites. Our evaluation includes 30 ESs, the computing capacity of each ES is distributed in [4, 6] GHz [24]. Their hardware reliability is distributed in $\mathbb{R}_e = [0.999 - 0.99999]$, following industry reports [46]. Each unit of computing resources consumes 1–10 units of cost. Besides, we set $\delta = 3$. These parameters serve as the default settings unless stated otherwise.

Metrics: To effectively evaluate our algorithms, we test them in various metrics, including the cost, approximation ratio [47], execution time, SFC acceptance ratio, number of backups, and maximum capacity violation. All data points are obtained by averaging over 20 runs.

Benchmark Algorithms: We compare our algorithms with existing solutions as follows.

- SBD-Off: This algorithm is a simplified version of the SBD algorithm, where only off-site backups are allowed.
- PG-Off: This algorithm is a simplified version of the PG algorithm, where only off-site backups are allowed.
- EI [15]: This method ensures SFC reliability by deploying and backing up VNF instances using off-site backups.
- Single [24]: The algorithm deploys and backs up VNF instances to ensure the reliability of each individual VNF, considering both software and hardware reliability.
- RABA [13]: The algorithm deploys and backs up each SFC based on its reliability requirements but ignores hardware reliability.
- Optimal: The optimal solution is derived by exhaustively exploring the solution space using the branch-and-bound technique [48].

6.2. Simulation result

In this subsection, we verify the effectiveness and superiority of our algorithms by comparing them with the optimal solution with existing solutions in different metrics.

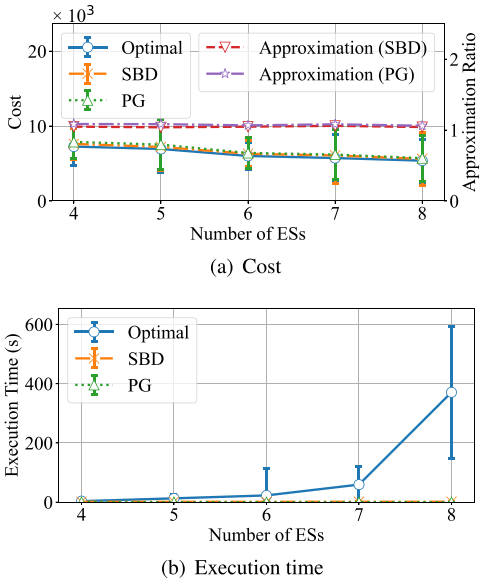


Fig. 3. Performance gap between our algorithm and optimal solution.

6.2.1. Compare with optimal solution

To evaluate the effectiveness of our algorithms, we compare them with the optimal solution in terms of cost, approximation ratio and execution time. We set $\mathcal{R} = 0.999999$ and $|\mathbb{V}| = 3$,[9] with results shown in Fig. 3.

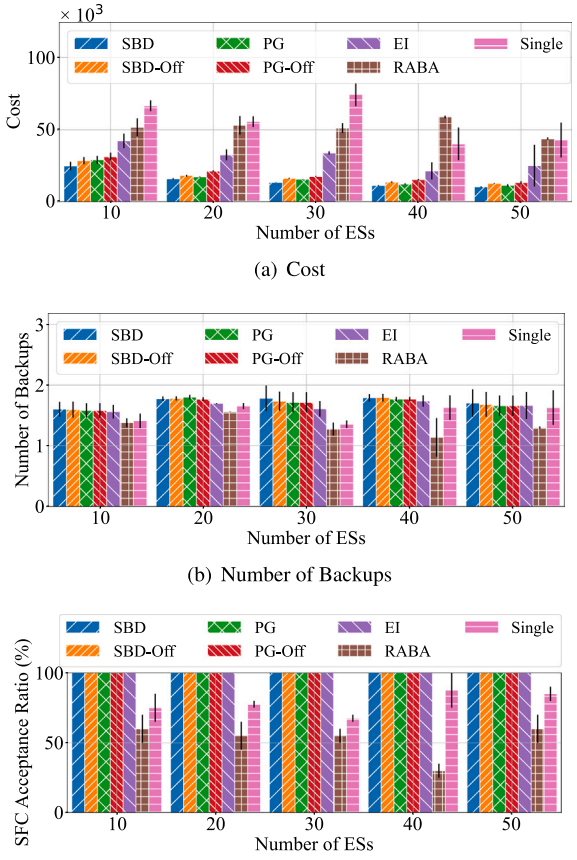
Fig. 3(a) illustrates that the cost of the SBD and PG algorithms closely approximates the optimal solution. In the worst case (7 ESs), their average costs are 6077.3 and 6200.0, respectively, compared to the optimal 5732.7, with standard deviations of 3077.9, 3759.8, and 3342.2. This indicates that our algorithms deviate from the optimal solution by only 8.2%. Additionally, as the number of ESs increases, costs decline due to the availability of more cost-effective and reliable ESs for backup and deployment. Moreover, the approximation ratio of our SBD algorithm and PG algorithm remains relatively small, not exceeding 1.082 and 1.06, respectively. This is clearly smaller than the approximation ratios in Theorems 4 and 6, consistent with our theorems.

Fig. 3(b) shows that our algorithms significantly outperform the optimal solution in execution time. While the optimal solution's execution time increases exponentially, reaching 370.8 s for 8 ESs, the SBD and PG algorithms complete in just 1.54 s and 3.01 ms, respectively. The rapid growth of the solution space—e.g., $4^{24} \approx 2.81 \times 10^{14}$ when $|\mathbb{E}| = 8$, $\delta = 3$, and $|\mathbb{V}| = 3$—leads to the exponential runtime of the branch-and-bound method, despite its efficiency in pruning search space. The exponential increase in execution time for exploring the optimal solution is consistent with our Theorem 1, demonstrating that solving the optimal solution for our RSBD problem is time-consuming. Thus, our algorithms achieve near-optimal results with significantly lower computational overhead.

6.2.2. Compare with existing solutions

To assess the superiority and robustness of our algorithms, we compare them with existing methods in terms of cost, number of backups, and SFC acceptance ratio under varying numbers of ESs, SFC requests, and reliability requirements. Rejected SFC requests incur a penalty cost of $\gamma = 4000$, set based on our evaluation results to reflect the impact of rejection. The results are presented in Figs. 4, 5, and 6.

[9] These parameters are fixed (as a typical SFC setup) to eliminate the impact of additional factors.

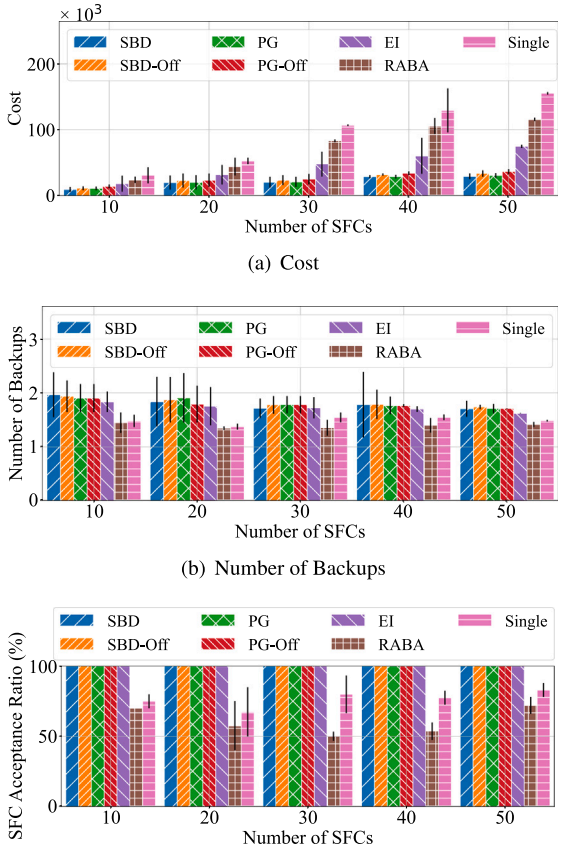


(a) Cost

(b) Number of Backups

(c) SFC Acceptance Ratio

Fig. 4. Performance under different number of ESs.



(a) Cost

(b) Number of Backups

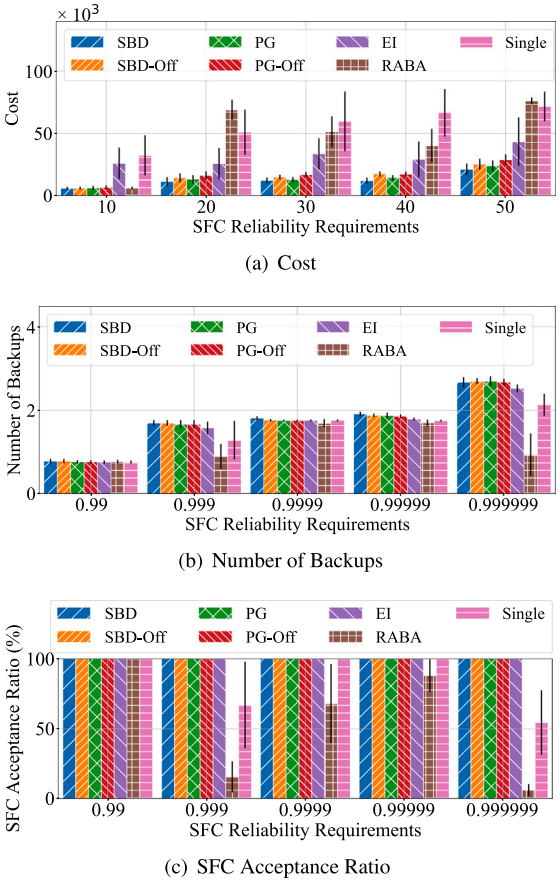
(c) SFC Acceptance Ratio

Fig. 5. Performance under different number of SFCs.

As shown in Fig. 4(a), our algorithms significantly reduce costs compared to existing approaches. For instance, with 30 ESs, the average costs for the SBD, SBD-Off, PG, PG-Off, EI, RABA, and Single algorithms are 13454, 16378.5, 15627, 17627, 33905, 51195, and 74395.5, respectively. This means that the SBD algorithm can save 60.3%, 73.7%, and 81.9% of the cost compared to the EI algorithm, RABA algorithm, and Single algorithm. This is because compared with the EI algorithm, the hybrid backup in the SBD algorithm provides a larger decision space and more opportunities to explore solutions that save costs. Moreover, in RABA and Single algorithms, many requests are rejected due to reliability violations, leading to high penalty costs. Besides, compared to SBD-Off and PG-Off, SBD and PG can save costs of 17.86% and 11.34%, respectively. This means that compared to off-site backup only, hybrid backup does bring significant resource cost savings. In addition, compared with heuristic-based PG algorithms, SBD algorithms can significantly reduce costs by exploring larger solution spaces by solving relaxation optimization problems. However, the SBD algorithm may face longer runtime and potential constraint violations. Note that even though only off-site backups are allowed in both SBD-off algorithm and PG-Off algorithm, it still has significantly lower cost relative to the EI algorithm thanks to our well-designed algorithm. Specifically, compared to the EI algorithm, our algorithm avoids deploying VNFs to high-cost ESs, even if the reliability of the SFC remains zero after adding functional instances to the SFC (e.g., the VNF components of the SFC have not yet all been deployed with a primary instance), which cannot be done with EI. This is consistent with our statement in the third paragraph of Section 5.1.1. This means that the superiority of our algorithm over existing algorithms is not only due to the hybrid backup mechanism, but also to the well-designed algorithm. Finally, as the number of ESs increases, the cost of the SBD algorithm gradually decreases, since more reliable and low-cost ESs are available to deploy and backup SFC requests.

Fig. 4(b) shows the number of backups for different algorithms under different number of ESs. The algorithms consume similar backups to deploy and back up SFC requests, among which SBD consumes more backups, which indicates that more low-cost and low-reliability ones are used to deploy SFC, which may bring lower costs despite consuming more backups. Besides, in the RABA and Single algorithms, fewer backups are consumed, which is because SFCs with higher reliability requirements are more likely to be rejected in them, because they ignore hardware reliability or are only customized for SFCs with a single VNF. This may lead to frequent SFC request rejections and low SFC acceptance ratios in RABA and Single algorithms, as shown in Fig. 4(c). What is more, SBD, SBD-Off, PG, PG-Off, and EI algorithms are customized for reliable SFC provision with multiple VNFs, and all considering the hardware reliability of ESs and the software reliability of VNFs, which can well meet the reliability requirements of SFCs.

Fig. 5 presents a comparison of our proposed algorithms with existing methods in terms of cost, backup count, and SFC acceptance ratio under varying SFC request numbers. As shown in Fig. 5(a), the SBD algorithm consistently achieves the lowest cost across different request numbers. Additionally, the resource consumption of all algorithms increases as the number of SFC requests grows, due to the fact that more SFC requests result in more primary or backup instances of their



(a) Cost

(b) Number of Backups

(c) SFC Acceptance Ratio

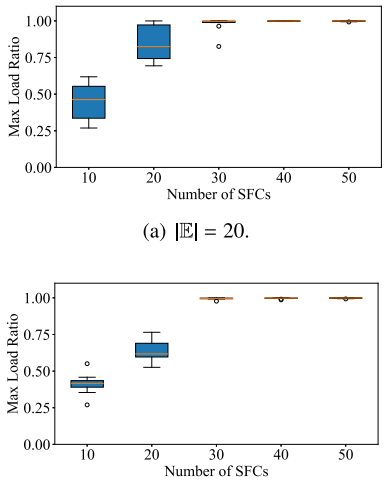
Fig. 6. Performance under different reliability requirements.

component VNFs needing to be deployed. Similar trends are observed in Figs. 5(b) and 5(c).

Fig. 6 evaluates the impact of SFC reliability requirements on cost, backup count, and SFC acceptance ratio. In Fig. 6(a), the costs of the SBD, PG, and EI algorithms increase with higher reliability requirements, as additional backups are needed to ensure service reliability. Fig. 6(b) shows a corresponding increase in backup count. Note that the RABA algorithm also has a low cost when the SFC reliability requirement is 0.99. This is because the RABA algorithm satisfies the reliability constraint when low reliability is required and is not penalized by the cost of constraint violation, and it is simultaneously able to deploy backups in a resource efficient manner. Meanwhile, Fig. 6(c) demonstrates that the acceptance ratio of the RABA and Single algorithms declines as reliability requirements rise. This is because RABA overlooks hardware reliability, potentially violating reliability constraints, particularly for stringent SFC requirements. Similarly, the Single algorithm focuses on ensuring individual VNF reliability, which can lead to mismatches between VNF and SFC reliability under strict constraints.

6.2.3. Capacity violation

Since the SBD algorithm follows a bi-criteria approximation approach, it may lead to slight capacity constraint violations. We now evaluate its maximum load ratio under varying numbers of ESs and SFC requests. The maximum load ratio is defined as $\psi = \max_{e\in\mathbb{E}} \frac{L_e}{K_e}$, where L_e represents the total computing resources allocated to SFCs on ES e. The results are illustrated in Fig. 7.



(a) $|\mathbb{E}| = 20$.

(b) $|\mathbb{E}| = 30$.

Fig. 7. Capacity occupancy ratio under different parameter settings.

As depicted in Fig. 7(a), the SBD algorithm results in only minor capacity violations. For instance, with 50 SFCs, the maximum violation rate ranges from 0.9924 to 1.0151, meaning that even in the worst case, the capacity is exceeded by just 1.51%. As the number of SFCs increases, the maximum load ratio also grows due to higher resource demands, yet it never surpasses 1.0151, which remains within an acceptable range. This limited over-allocation is reasonable in practical deployments, as controlled resource over-provisioning is commonly used to enhance utilization. Moreover, backup VNF instances are only activated upon primary instance failure, making moderate over-allocation a practical trade-off. Similar trends are observed in Fig. 7(b), where increasing the number of ESs from 20 to 30 reduces the maximum load ratio. This occurs because additional ESs provide more deployment flexibility, alleviating excessive resource allocation. Overall, our findings confirm that the SBD algorithm only introduces slight capacity violations, further supporting Theorem 4, which remains manageable in real-world scenarios.

7. Conclusion

This paper addresses the cost-effective hybrid backup and deployment of SFCs to meet reliability requirements while minimizing resource consumption. Considering key factors such as hardware and software reliability and resource charging heterogeneity, we establish a mathematical model linking backup and deployment decisions to these factors, formulate it as an integer nonlinear programming problem, and analyze its complexity. To solve this, we propose a bi-criteria approximation algorithm based on convex relaxation and rounding, ensuring a bounded approximation ratio while allowing limited capacity violations, making it suitable for scenarios permitting resource over-allocation. To handle scenarios where resource over-allocation is prohibited, we devise a priority-guided algorithm with rigorous theoretical guarantees based on our insights, which prioritizes deploying backups for VNFs with the lowest reliability on ESs that bring higher reliability improvements and lower charges. Finally, we conducted extensive evaluations to verify that our algorithms significantly outperform existing algorithms in resource cost.

CRediT authorship contribution statement

Yue Zeng: Writing – original draft, Validation, Methodology, Investigation, Formal analysis. **Pan Li:** Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Shanshan Lin:** Software, Methodology, Data curation. **Bin Tang:** Validation, Formal analysis. **Xiaoliang Wang:** Validation, Methodology. **Zhihao Qu:** Investigation, Formal analysis. **Baoliu Ye:** Validation, Supervision. **Song Guo:** Validation, Investigation. **Junlong Zhou:** Validation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China[http://dx.doi.org/10.13039/501100001809] under Grants 62402226, 62172224, 62441225, in part by the Natural Science Foundation of Jiangsu Province, China under Grants BK20241453, BK20220138, in part by the Fundamental Research Funds for the Central Universities, China under Grants 30924010817, 30922010318, and in part by the Open Project Program of the National Key Laboratory for Novel Software Technology under Grant KFKT2024B17.

Appendix A. Proof of Theorem 2

To facilitate understanding of the proof of this theorem, we give the definition of convex optimization [49], as follows.

Definition 3. A convex optimization problem is a problem where all of the constraints are convex functions, and the objective is a convex function if minimizing, or a concave function if maximizing.

As shown in R-RSBD problem, the objective function and constraints are linear, except for constraints (2), (3), and (4). To prove that RSBD problem is convex, we only need to prove that constraints (2), (3), (4) are convex.

Generally speaking, equations with cumulative multiplication terms are difficult to analyze and solve. Besides, logarithmic functions are obviously monotonically increasing. Therefore, if we take the logarithm at both ends of Eq. (4), the inequality still holds. Then Eq. (4) can be replaced with

$$\sum_{v\in\mathbb{V}} \ln R_v \geq \ln \mathcal{R}. \tag{A.1}$$

Combining Eqs. (2), (3), (A.1), we have

$$\sum_{v\in\mathbb{V}} \ln(1-\prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{n_{v,e}}))) \geq \ln \mathcal{R}. \tag{A.2}$$

$$\Psi(\boldsymbol{x}) = \mathcal{R} - \prod_{v\in\mathbb{V}}(1-\prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{x_{v,e}}))), \tag{A.3}$$

Now, we define a function

$$\Psi(\boldsymbol{x}) = \ln \mathcal{R} - \sum_{v\in\mathbb{V}} \ln(1-\prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{x_{v,e}}))), \tag{A.4}$$

where $\boldsymbol{x} = \{x_{v,e} \in \{0,1,\ldots,\delta\}, \forall v \in \mathbb{V}, e \in \mathbb{E}\}$. To prove that constraints (2), (3), and (4) are convex, we only need to prove that $\Psi(\boldsymbol{x})$ is convex.

To facilitate analysis, we define another function

$$f(\boldsymbol{x}_v) = -\ln(1-\prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{x_{v,e}}))), \tag{A.5}$$

where $\boldsymbol{x}_v = \{x_{v,e}, \forall e \in \mathbb{E}\}, \forall v \in \mathbb{V}$.

Based on Eq. (A.4), (A.5), we have

$$\Psi(\boldsymbol{x}) = \ln \mathcal{R} + \sum_{v\in\mathbb{V}} f(\boldsymbol{x}_v). \tag{A.6}$$

Since $\ln \mathcal{R}$ is a constant and the accumulation of convex functions is still convex, once $f(\boldsymbol{x}_v)$ is convex, $\Psi(\boldsymbol{x})$ is convex. Therefore, we next need to prove that $f(\boldsymbol{x}_v)$ is convex. However, $f(\boldsymbol{x}_v)$ is complicated and hinders analysis. To facilitate the proof, we split it by defining the following functions

$$\begin{aligned} h(z) &= -\ln(z), \\ g(\boldsymbol{y}) &= 1-\prod_{e\in\mathbb{E}}(1-y_e), \\ I(u) &= r_e(1-(1-r_v)^u), \end{aligned} \tag{A.7}$$

where $\boldsymbol{y} = \{y_e, \forall e \in \mathbb{E}\}$.

Next, we prove that $f(\boldsymbol{x}_v)$ is convex by proving the convexity of the above three functions. To pave the way for the proof, we give the following definitions [49] of convexity preservation and concavity preservation properties for composite functions as follows.

Definition 4. $f = h(g(\boldsymbol{y}))$ is convex if h is convex and nonincreasing, and g is concave.

Definition 5. $f' = g(I(u))$ is convex if g is concave and nondecreasing, and I is concave.

Lemma 1. *$h(z)$ is convex and nonincreasing.*

Proof. As in Literature [49], the basic function $\ln z$ is concave. Besides, the convex function is the negative of the concave function. Obviously, $h(z)$ is convex. Moreover, $\ln z$ is increasing, and the negative of the increasing function is a decreasing function. Therefore, $h(z)$ is nonincreasing. □

Lemma 2. *$g(\boldsymbol{y})$ is concave and nondecreasing.*

Proof. Next, we prove the concavity of $g(\boldsymbol{y})$ by taking the second-order partial derivative for it as

$$\frac{\partial^2 g}{\partial y_i \partial y_j} = \begin{cases} 0, & \text{if } i=j, i\in\mathbb{E}, j\in\mathbb{E} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_i)\cdot(1-y_j)}, & \text{otherwise.} \end{cases} \tag{A.8}$$

Therefore, the Hessian matrix of $g(\boldsymbol{y})$ is $H(g) =$

$$\begin{bmatrix} 0 & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})\cdot(1-y_{e_1})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|S|}})\cdot(1-y_{e_1})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_2})} & 0 & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})\cdot(1-y_{e_2})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_3})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})\cdot(1-y_{e_3})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})\cdot(1-y_{e_3})} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_\mathbb{E}})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_{e_2})}{(1-y_{e_2})\cdot(1-y_{e_{|\mathbb{E}|}})} & \cdots & 0 \end{bmatrix},$$

where e_i denotes the ith element in $\mathbb{E}$.

Then, we have $|H(g)| =$

$$\begin{vmatrix} 0 & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})\cdot(1-y_{e_1})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})\cdot(1-y_{e_1})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_2})} & 0 & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})\cdot(1-y_{e_2})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_3})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})\cdot(1-y_{e_3})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})\cdot(1-y_{e_3})} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})\cdot(1-y_{e_{|\mathbb{E}|}})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_{e_2})}{(1-y_{e_2})\cdot(1-y_{e_{|\mathbb{E}|}})} & \cdots & 0 \end{vmatrix}$$

$$= \frac{1}{\prod_{e\in\mathbb{E}}(1-y_e)} \cdot \begin{vmatrix} 0 & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & 0 & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_{e_2})}{(1-y_{e_2})} & \cdots & 0 \end{vmatrix}$$

$$= \frac{|\mathbb{E}|-1}{\prod_{e\in\mathbb{E}}(1-y_e)} \cdot \begin{vmatrix} \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & 0 & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_{e_2})}{(1-y_{e_2})} & \cdots & 0 \end{vmatrix}$$

$$= \frac{|\mathbb{E}|-1}{\prod_{e\in\mathbb{E}}(1-y_e)} \cdot \begin{vmatrix} \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_1})} & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & \frac{-\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \\ 0 & \frac{\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_2})} & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \frac{\prod_{e\in\mathbb{E}}(1-y_e)}{(1-y_{e_{|\mathbb{E}|}})} \end{vmatrix}$$

$$= (1-|\mathbb{E}|)\cdot\left(\prod_{e\in\mathbb{E}}(1-y_e)\right)^{|\mathbb{E}|-2}$$

Since the values of reliability are distributed in $[0,1]$, then $R_{v,e} \le 1$, and then $y_e \le 1$. Besides, ESs must exist for deploying services, ie, $|\mathbb{E}| \ge 1$, then we have $|H(g)| \le 0$. Thus, $g(\boldsymbol{x})$ is concave.

Besides, based on Eq. (A.7), the first-order partial derivative of $g(\boldsymbol{y})$ can be calculated as

$$\frac{\partial g(\boldsymbol{y})}{\partial y_e} = \prod_{e'\neq e}(1-y_{e'}) \ge 0, \tag{A.9}$$

where the second equation holds because $y_{e'} = I(u), 1 \ge I(u) = r_e(1-(1-r_v)^u) \ge 0$, due to $1 \ge r_e \ge 0$, $1 \ge r_v \ge 0$, and $\delta \ge u \ge 0$. Thus, $g(\boldsymbol{y})$ is concave and nondecreasing. □

Lemma 3. *$I(z)$ is concave.*

Proof. According to Eq. (A.7), the second order derivative of $I''(u)$ can be calculated as

$$I''(u) = -r_e\cdot(\ln(1-r_v))^2\cdot(1-r_v)^u \le 0, \tag{A.10}$$

where the second inequality holds because $1 \ge r_e \ge 0, 1 \ge r_v \ge 0$.

Thus, $I(z)$ is concave. □

According to Definitions 4, 5, and Lemmas 1, 2, 3, the constraints composed of constraints (2), (3), (4) are convex, therefore R-RSBD problem is convex.

Appendix B. Proof of Theorem 3

We prove this theorem by proving the following two lemmas.

Lemma 4. *SBD algorithm approximates the optimal solution by factor $\lfloor \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}})+1\rfloor \frac{\tau_{max}}{\tau_{min}}$.*

Proof. First, we prove that our algorithm does not lose any feasible solution as follows. In SBD algorithm, we reject an SFC request only when R-RSBD problem is infeasible. Since R-RSBD problem is derived from the relaxation of RSBD problem, only when RSBD problem is infeasible, R-RSBD problem may be infeasible. Thus, our algorithm rejects an SFC request only when RSBD problem is infeasible.

Next, we prove that SBD algorithm approximates the optimal solution when an SFC request is accepted as follows.

To facilitate the following proof, we define a variable $n'_{v,e}$ to denote the backup and deployment decision for VNF v before its last instance is added. Then, we have

$$\sum_{e\in\mathbb{E}} n'_{v,e} + 1 = \sum_{e\in\mathbb{E}} n_{v,e}, \forall v \in \mathbb{V}. \tag{B.1}$$

We also define R'_v as the reliability of VNF v before its last backup instance is added. Then, based on Eq. (2), (3), we have

$$R'_v = 1 - \prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{n'_{v,e}})). \tag{B.2}$$

Based on Eqs. (B.2), $\xi = \min_{v\in\mathbb{V}}\{r_v\}, \eta = \min_{e\in\mathbb{E}}\{r_e\}$, we have

$$\begin{aligned} 1-R'_v &= \prod_{e\in\mathbb{E}}(1-r_e(1-(1-r_v)^{n'_{v,e}})) \\ &\le \prod_{e\in\mathbb{E}}(1-\eta(1-(1-\xi)^{n'_{v,e}})). \end{aligned} \tag{B.3}$$

Since $n'_{v,e} \in \{0,1,\ldots,\delta\}$, based on Eq. (B.3), we have

$$\begin{aligned} 1-R'_v &\le \prod_{e\in\mathbb{E}}(1-\eta(1-(1-\xi)^{n'_{v,e}})) \le \prod_{e\in\mathbb{E}}(1-\xi\eta)^{\frac{n'_{v,e}}{\delta}} \\ &= (1-\xi\eta)^{\frac{\sum_{e\in\mathbb{E}} n'_{v,e}}{\delta}}, \end{aligned} \tag{B.4}$$

where the second inequality is because when $n'_{v,e} = 0$, it clearly holds, while when $n'_{v,e} \in \{1,\ldots,\delta\}$, So $1-\eta(1-(1-\xi)^{n'_{v,e}}) \le 1-\eta(1-(1-\xi)) = 1-\eta\xi \le (1-\eta\xi)^{\frac{n'_{v,e}}{\delta}}$.

Proposition 2. *Once $\min_{v\in\mathbb{V}}\{R_v\} \ge \mathcal{R}^{\frac{1}{|\mathbb{V}|}}$, then the SFC reliability is satisfied.*

Proof. Based on Eq. (1), we have

$$R = \prod_{v\in\mathbb{V}} R_v \ge \prod_{v\in\mathbb{V}} \min_{v'\in\mathbb{V}}\{R_{v'}\} \tag{B.5}$$

When $\min_{v\in\mathbb{V}}\{R_v\} \ge \mathcal{R}^{\frac{1}{|\mathbb{V}|}}$, based on (B.5), we have

$$R \ge \prod_{v\in\mathbb{V}} \min_{v'\in\mathbb{V}}\{R_{v'}\} \ge \prod_{v\in\mathbb{V}} \mathcal{R}^{\frac{1}{|\mathbb{V}|}} = \mathcal{R}, \tag{B.6}$$

which clearly satisfies the SFC reliability requirement. □

In the RSBD algorithm, backups are iteratively added to the VNF with the lowest reliability until the SFC meets its reliability requirement. When an additional instance is required, the selected VNF is the least reliable, and the SFC's reliability remains unsatisfied. According to Proposition 2, its reliability is below $\mathcal{R}^{\frac{1}{|\mathbb{V}|}}$. That is,

$$R'_v < \mathcal{R}^{\frac{1}{|\mathbb{V}|}}, \forall v \in \mathbb{V}, \tag{B.7}$$

Since $\xi = \min_{v\in\mathbb{V}}\{r_v\}, \eta = \min_{e\in\mathbb{E}}\{r_e\}$, we have $1 \ge 1-\xi\eta \ge 0$. By taking the logarithm of Eq. (B.4), we have

$$\frac{\sum_{e\in\mathbb{E}} n'_{v,e}}{\delta} \le \log_{1-\xi\eta}(1-R'_v) \le \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}), \tag{B.8}$$

where the second inequality is obtained by Eq. (B.7).

Obviously, based on Eqs. (B.1), (B.8), we have

$$\sum_{e\in\mathbb{E}} n_{v,e} = \sum_{e\in\mathbb{E}} n'_{v,e} + 1 \le \delta \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1. \tag{B.9}$$

Since $n_{v,e} \in \{0,1,\dots,\delta\}$, $\sum_{e\in\mathbb{E}} n_{v,e}$ is also a integer. Then,

$$\sum_{e\in\mathbb{E}} n_{v,e} \le \lfloor \delta \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor. \tag{B.10}$$

Based on the above analysis, we can obtain the solution of RSBD algorithm as

$$\begin{aligned} SOL_R &= \sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_e \kappa_v n_{v,e} \le \sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_{max}\kappa_v n_{v,e} \\ &\le \tau_{max}\cdot \sum_{v\in\mathbb{V}} (\kappa_v \cdot \sum_{e\in\mathbb{E}} n_{v,e}) \\ &\le \tau_{max}\cdot \lfloor \delta \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor \cdot \sum_{v\in\mathbb{V}} \kappa_v, \end{aligned} \tag{B.11}$$

where the second inequality is due to $\tau_{max} = \max_{v\in\mathbb{V}}\{\tau_v\}$ and the fourth inequality is due to Eq. (B.10).

The optimal solution of RSBD problem can be calculated as

$$\begin{aligned} OPT_R &= \sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_e \kappa_v n^*_{v,e} \ge \sum_{v\in\mathbb{V}}\sum_{e\in\mathbb{E}} \tau_{min}\kappa_v n^*_{v,e} \\ &= \tau_{min}\sum_{v\in\mathbb{V}} (\kappa_v \sum_{e\in\mathbb{E}} n^*_{v,e}) \ge \tau_{min}\sum_{v\in\mathbb{V}} \kappa_v \end{aligned} \tag{B.12}$$

where $n^*_{v,e}$ indicates the optimal SFC backup and deployment decision. The second inequality is due to $\tau_{min} = \min_{v\in\mathbb{V}}\{\tau_v\}$ and the fourth inequality is due to Eq. (5).

Based on Eqs. (B.11), (B.12), the approximate ratio of RSBD algorithm is

$$\begin{aligned} \alpha &= \frac{SOL_R}{OPT_R} \\ &\le \frac{\tau_{max}\cdot \lfloor \delta \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor \cdot \sum_{v\in\mathbb{V}} \kappa_v}{\tau_{min}\cdot \sum_{v\in\mathbb{V}} \kappa_v} \quad \square \\ &= \lfloor \delta \log_{1-\xi\eta}(1-\mathcal{R}^{\frac{1}{|\mathbb{V}|}}) + 1 \rfloor \cdot \frac{\tau_{max}}{\tau_{min}}. \end{aligned} \tag{B.13}$$

Lemma 5. *SBD algorithm can guarantee that the capacity constraint will not be violated by a factor* $1 + \frac{\sum_{v\in\mathbb{V}}\kappa_v}{K_e}$*, while other constraints are not violated, including SFC completeness, reliability, and hybrid backup constraints.*

Proof. We first prove that SBD algorithm will not exceed the capacity constraint by factor $1 + \frac{\sum_{v\in\mathbb{V}}\kappa_v}{K_e}$, as follows.

Since $\widetilde{n}_{v,e}$ is the optimal solution to R-RSBD problem, we have

$$\sum_{v\in\mathbb{V}} \kappa_v \widetilde{n}_{v,e} \le K_e. \tag{B.14}$$

Besides, as shown in SBD algorithm, $n_{v,e}$ is the result of rounding down or rounding up $\widetilde{n}_{v,e}$. That is,

$$n_{v,e} \in \{\lfloor \widetilde{n}_{v,e} \rfloor, \lceil \widetilde{n}_{v,e} \rceil\}. \tag{B.15}$$

Based on Eqs. (B.14), (B.15), we have

$$\begin{aligned} \sum_{v\in\mathbb{V}} \kappa_v n_{v,e} &\le \sum_{v\in\mathbb{V}} \kappa_v(\widetilde{n}_{v,e}+1) = \sum_{v\in\mathbb{V}} \kappa_v \widetilde{n}_{v,e} + \sum_{v\in\mathbb{V}} \kappa_v \\ &\le K_e + \sum_{v\in\mathbb{V}} \kappa_v. \end{aligned} \tag{B.16}$$

Obviously, the capacity occupancy ratio can be calculated as

$$\frac{\sum_{v\in\mathbb{V}} \kappa_v n_{v,e}}{K_e} \le \frac{K_e + \sum_{v\in\mathbb{V}} \kappa_v}{K_e} = 1 + \frac{\sum_{v\in\mathbb{V}} \kappa_v}{K_e}. \tag{B.17}$$

Next, we prove that SBD algorithm will not violate the hybrid backup constraint, reliability constraint and SFC completeness constraint.

As shown in Algorithm 1, $\widetilde{n}_{v,e}$ is obtained by solving R-RSBD problem, so $0 \le \widetilde{n}_{v,e} \le \delta$. The variable $n_{v,e}$ is initialized to 0 and may be reassigned to $\lfloor \widetilde{n}_{v,e} \rfloor$ or $\lceil \widetilde{n}_{v,e} \rceil$. Obviously, $n_{v,e} \in \{0,1,\dots,\delta\}$, which satisfies the hybrid backup constraint.

Next, we prove that RSBD algorithm satisfies the reliability constraint when an SFC request is accepted, as follows. As shown in Algorithm 1, our algorithm accepts an SFC request when R-RSBD problem has a feasible solution. The algorithm rounds up the fractional solution obtained by solving R-RSBD problem until the reliability is satisfied. In the worst case, we round up all the fractional solution in R-RSBD algorithm. Then, we have

$$\begin{aligned} R &= 1 - \prod_{e\in\mathbb{E}} (1 - r_e(1-(1-r_v)^{n_{v,e}})) \\ &= 1 - \prod_{e\in\mathbb{E}} (1 - r_e(1-(1-r_v)^{\lceil n_{v,e} \rceil})) \\ &\ge 1 - \prod_{e\in\mathbb{E}} (1 - r_e(1-(1-r_v)^{n_{v,e}})) = \widetilde{R} \ge \mathcal{R}, \end{aligned} \tag{B.18}$$

where $\widetilde{n}_{v,e}$ denotes the optimal solution obtained by solving R-RSBD problem, and $\widetilde{R}$ denotes the corresponding reliability. Therefore, SBD algorithm meets the reliability requirement when an SFC request is accepted.

Finally, we show that the SFC completeness constraint is satisfied. Once a VNF $v \in \mathbb{V}$ is not deployed with an instance, we have $n_{v,e} = 0, \forall e \in \mathbb{E}$. In this case, $R_v = 1 - \prod_{e\in\mathbb{E}}(1 - r_e(1-(1-r_v)^0)) = 0$. Further $R = \prod_{v\in\mathbb{V}} R_v = 0$, which violates the reliability constraint. That is, $R < \mathcal{R}$, which contradicts our proof of reliability satisfaction above. Therefore, the SFC completeness constraint is always satisfied as long as the reliability constraint is satisfied. $\square$

Obviously, the theorem is proved by the above two lemmas.

Data availability

No data was used for the research described in the article.

References

[1] Y.C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, Mobile edge computing—A key technology towards 5G, ETSI White Pap. 11 (11) (2015) 1–16.

[2] Y. Zeng, J. Zhou, B. Ye, Z. Qu, S. Guo, T. Gong, P. Li, ExpertDRL: Request dispatching and instance configuration for serverless edge inference with foundation models, IEEE Trans. Mob. Comput. (2025) 1–16.

[3] J. Zhou, X. Hou, Y. Zeng, P. Cong, W. Jiang, S. Guo, Quality of experience and reliability-aware task offloading and scheduling for multi-user mobile-edge computing systems, IEEE Trans. Serv. Comput. (2025) 1–14.

[4] Y. Zeng, B. Ye, B. Tang, S. Lu, F. Xu, S. Guo, Z. Qu, Mobility-aware proactive flow setup in software-defined mobile edge networks, IEEE Trans. Commun. 71 (3) (2023) 1549–1563.

[5] P. Li, G. Liu, S. Guo, Y. Zeng, Traffic-aware efficient consistency update in NFV-enabled software defined networking, Comput. Netw. 228 (2023) 109755.

[6] V.R. Chintapalli, R. Partani, B.R. Tamma, et al., Energy efficient and delay aware deployment of parallelized service function chains in NFV-based networks, Comput. Netw. 243 (2024) 110289.

[7] P. Vidhya, K. Subashini, R. Sathishkannan, S. Gayathri, Dynamic network slicing based resource management and service aware virtual network function (VNF) migration in 5G networks, Comput. Netw. 259 (2025) 111064.

[8] 3GPPTS 22.261, Service Requirements for the 5G System (Release 20), Tech. Rep., Technical Specification Group Services and System Aspects, 2025.

[9] J. Sherry, P.X. Gao, S. Basu, A. Panda, A. Krishnamurthy, C. Maciocco, M. Manesh, J. Martins, S. Ratnasamy, L. Rizzo, et al., Rollback-recovery for middleboxes, in: ACM SIGCOMM, 2015, pp. 227–240.

[10] M. Ghaznavi, E. Jalalpour, B. Wong, R. Boutaba, A.J. Mashtizadeh, Fault tolerant service function chaining, in: ACM SIGCOMM, 2020, pp. 198–210.

[11] J. Fan, M. Jiang, C. Qiao, Carrier-grade availability-aware mapping of service function chains with on-site backups, in: IEEE/ACM IWQoS, 2017, pp. 1–10.

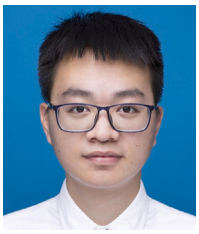
[12] J. Fan, C. Guan, Y. Zhao, C. Qiao, Availability-aware mapping of service function chains, in: IEEE INFOCOM, 2017, pp. 1–9.

[13] J. Zhang, Z. Wang, C. Peng, L. Zhang, T. Huang, Y. Liu, RABA: Resource-aware backup allocation for a chain of virtual network functions, in: IEEE INFOCOM, 2019, pp. 1918–1926.

[14] X. Shang, Y. Huang, Z. Liu, Y. Yang, Reducing the service function chain backup cost over the edge and cloud by a self-adapting scheme, in: IEEE INFOCOM, 2020, pp. 2096–2105.

[15] Y. Zeng, Z. Qu, S. Guo, B. Ye, J. Zhang, J. Li, B. Tang, SafeDRL: Dynamic microservice provisioning with reliability and latency guarantees in edge environments, IEEE Trans. Comput. 73 (1) (2024) 235–248.

[16] R. Cohen, L. Lewin-Eytan, J.S. Naor, D. Raz, Near optimal placement of virtual network functions, in: IEEE INFOCOM, 2015, pp. 1346–1354.
[17] J. Martín-Pérez, F. Malandrino, C.-F. Chiasserini, C.J. Bernardos, OKpi: All-KPI network slicing through efficient resource allocation, in: IEEE INFOCOM, 2020, pp. 804–813.
[18] L. Yala, P.A. Frangoudis, A. Ksentini, Latency and availability driven VNF placement in a MEC-NFV environment, in: IEEE GLOBECOM, 2018, pp. 1–7.
[19] R. Potharaju, N. Jain, Demystifying the dark side of the middle: A field study of middlebox failures in datacenters, in: ACM IMC, 2013, pp. 9–22.
[20] Y. Kanizo, O. Rottenstreich, I. Segall, J. Yallouz, Optimizing virtual backup allocation for middleboxes, IEEE/ACM Trans. Netw. 25 (5) (2017) 2759–2772.
[21] C. Peng, D. Zheng, Y. Zhong, X. Cao, Off-site protection against service function forwarder failures in NFV, Comput. Netw. 221 (2023) 109510.
[22] D. Zheng, X. Cao, Provably efficient service function chain embedding and protection in edge networks, IEEE/ACM Trans. Netw. 33 (1) (2025) 178–193.
[23] D. Zheng, H. Fang, S. Cao, Y. Zhong, X. Cao, Towards resources optimization in deploying service function chains with shared protection, Comput. Netw. 248 (2024) 110494.
[24] J. Li, W. Liang, M. Huang, X. Jia, Reliability-aware network service provisioning in mobile edge-cloud networks, IEEE Trans. Parallel Distrib. Syst. 31 (7) (2020) 1545–1558.
[25] X. Liu, D. Zheng, H. Xing, L. Feng, C. Peng, X. Cao, A cost-provable solution for reliable in-network computing-enabled services deployment, Comput. Netw. 257 (2025) 110997.
[26] Y. Zeng, Z. Qu, S. Guo, B. Tang, B. Ye, J. Li, J. Zhang, RuleDRL: Reliability-aware SFC provisioning with bounded approximations in dynamic environments, IEEE Trans. Serv. Comput. 16 (5) (2023) 3651–3664.
[27] M. Chiosi, D. Clarke, P. Willis, A. Reid, J. Feger, M. Bugenhagen, W. Khan, M. Fargano, C. Cui, H. Deng, et al., Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action, introductory white paper, in: SDN and OpenFlow World Congress, vol. 48, 2012, pp. 1–16.
[28] G. NFV, et al., Network functions virtualisation (NFV); architectural framework, NFV ISG 2 (2) (2013) V1.
[29] M. Huang, W. Liang, X. Shen, Y. Ma, H. Kan, Reliability-aware virtualized network function services provisioning in mobile edge computing, IEEE Trans. Mob. Comput. 19 (11) (2020) 2699–2713.
[30] W. Liang, Y. Ma, W. Xu, Z. Xu, X. Jia, W. Zhou, Request reliability augmentation with service function chain requirements in mobile edge computing, IEEE Trans. Mob. Comput. Early Access (2021).
[31] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, et al., ONOS: Towards an open, distributed SDN OS, in: HotSDN, 2014, pp. 1–6.
[32] N. Katta, H. Zhang, M. Freedman, J. Rexford, Ravana: Controller fault-tolerance in software-defined networking, in: ACM SIGCOMM SOSR, 2015, pp. 1–12.
[33] A. Panda, W. Zheng, X. Hu, A. Krishnamurthy, S. Shenker, SCL: Simplifying distributed SDN control planes, in: USENIX NSDI, 2017, pp. 329–345.
[34] Faisal, Where are edge data centers located? 2025, URL https://telcocloudbridge.com/blog/where-are-edge-data-centers-located/.
[35] H. Zhang, S. Huang, M. Xu, D. Guo, X. Wang, V.C. Leung, W. Wang, How far have edge clouds gone? A spatial-temporal analysis of edge network latency in the wild, in: IEEE/ACM IWQoS, 2023, pp. 1–10.
[36] A. Bremler-Barr, Y. Harchol, D. Hay, OpenBox: A software-defined framework for developing, deploying, and managing network functions, in: ACM SIGCOMM, 2016, pp. 511–524.
[37] W. Haeffner, J. Napper, M. Stiemerling, D. Lopez, J. Uttaro, Service Function Chaining Use Cases in Mobile Networks, IETF, 2019, Internet-Draft, [Online] URL https://datatracker.ietf.org/doc/html/draft-ietf-sfc-use-case-mobility-09.
[38] Y. Yang, L. Zhao, Y. Li, H. Zhang, J. Li, M. Zhao, X. Chen, K. Li, INFless: A native serverless system for low-latency, high-throughput inference, in: Proceedings of ACM ASPLOS, 2022, pp. 768–781.
[39] S. Yang, F. Li, S. Trajanovski, R. Yahyapour, X. Fu, Recent advances of resource allocation in network function virtualization, IEEE Trans. Parallel Distrib. Syst. 32 (2) (2021) 295–314.
[40] B. Han, V. Gopalakrishnan, G. Kathirvel, A. Shaikh, On the resiliency of virtual network functions, IEEE Commun. Mag. 55 (7) (2017) 152–157.
[41] X. Jin, H.H. Liu, R. Gandhi, S. Kandula, R. Mahajan, M. Zhang, J. Rexford, R. Wattenhofer, Dynamic scheduling of network updates, ACM SIGCOMM Comput. Commun. Rev. 44 (4) (2014) 539–550.
[42] D.P. Williamson, D.B. Shmoys, The Design of Approximation Algorithms, Cambridge University Press, 2011.
[43] D.B. Shmoys, É. Tardos, Approximation algorithm for generalized assignment problem, Math. Program. 62 (1) (1993) 461–474.
[44] Y. Nesterov, A. Nemirovskii, Interior Point Polynomial Methods in Convex Programming, SIAM, 1994.
[45] M. Ehrgott, Multicriteria Optimization, vol. 491, Springer Science & Business Media, 2005.
[46] C. Cérin, C. Coti, P. Delort, F. Diaz, M. Gagnaire, Q. Gaumer, N. Guillaume, J. Lous, S. Lubiarz, J. Raffaelli, et al., Downtime Statistics of Current Cloud Solutions, Tech. Rep., International Working Group on Cloud Computing Resiliency, 2014.
[47] D. Zheng, G. Shen, Y. Li, X. Cao, B. Mukherjee, Service function chaining and embedding with heterogeneous faults tolerance in edge networks, IEEE Trans. Netw. Serv. Manag. 20 (3) (2022) 2157–2171.
[48] S. Boyd, J. Mattingley, Branch and Bound Methods, vol. 2006, Citeseer, 2007, p. 07.
[49] S. Boyd, S.P. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.



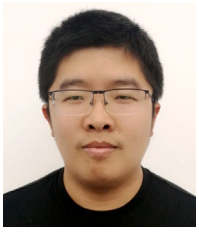
Yue Zeng received the Ph.D degree from Nanjing University. He is currently an associate professor in Nanjing University of Science and Technology. He has published more than a dozen papers in top journals and conferences, including IEEE Transactions on Computers, IEEE Transactions on Services Computing, IEEE Transactions on Mobile Computing, IEEE Transactions on Communications, IEEE Transactions on Cloud Computing and IEEE CVPR. His research interests include edge intelligence, deep reinforcement learning, machine learning training and inference, federated learning, and serverless computing.

Pan Li received the M.S. degree in the department of electronic information engineering from Southwest University, Chongqing, China, in 2019. She is currently working toward the Ph.D. degree in computer science and technology, Southwest University. Her research interests include network functions virtualization, software defined networking, and edge computing.

Shanshan Lin received the BEng degree in Computer Science and Technology from Guizhou University, Guizhou, China, in 2025. She is currently pursuing the M. degree at the School of Computer Science and Engineering, Nanjing University of Science and Technology. Her research interests include edge computing.

Bin Tang (Member, IEEE) received his B.S. and Ph.D. degree in computer science from Nanjing University, Nanjing, China, in 2007, and 2014, respectively. He was an assistant researcher at Nanjing University from 2014 to 2020, and also a research fellow at The Hong Kong Polytechnic University in 2019. He is currently a professor at Hohai University. His research interests lie in the area of communications, network coding, and distributed computing.

Xiaoliang Wang (Member, IEEE) received the Ph.D. degree from the Graduate School of Information Sciences, Tohoku University, Japan. From 2010 to 2014, he was an Assistant Professor with the Department of Computer Science and Technology, Nanjing University, China, where he is currently an Associate Professor. He has published more than 30 technical articles at premium international journals and conferences, including the IEEE TIT, IEEE TCOM, the IEEE INFOCOM, ACM SIGCOMM, USENIX NSDI, USENIX ATC, and USENIX FAST. His research interests include cloud computing, data networking, distributed systems, enterprise and data center networks.



Zhihao Qu (Member, IEEE) received his B.S. and Ph.D. degree in computer science from Nanjing University, Nanjing, China, in 2009, and 2018, respectively. He is currently an assistant researcher in the College of Computer and Information at Hohai University. His research interests are mainly in the areas of wireless networks, edge computing, and distributed machine learning.

Baoliu Ye (Member, IEEE) is a full professor at Department of Computer Science and Technology, Nanjing University. He received his Ph.D. in computer science from Nanjing University, China in 2004. He served as a visiting researcher of the University of Aizu, Japan from March 2005 to July 2006, and the Dean of School of Computer and Information, Hohai University since January 2018. His current research interests mainly include distributed systems, cloud computing, wireless networks with over 70 papers published in major conferences and journals. Prof. Ye served as the TPC co-chair of HotPOST12, Hot-POST11, P2PNet10. He is the regent of CCF, the SecretaryGeneral of CCF Technical Committee of Distributed Computing and Systems, and a member of IEEE.

Song Guo (Fellow, IEEE) is currently a full professor with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology. He is also the Changjiang chair professor awarded by the Ministry of Education of China. His research interests include edge AI, mobile computing, and distributed systems. He has been recognized as a highly cited researcher (Web of Science) and was the recipient of more than 14 best paper awards from IEEE/ACM conferences, journals, and technical committees. He is the editor-in-chief of the IEEE Open Journal of the Computer Society. He was on the IEEE Communications Society Board of Governors, IEEE Computer Society Fellow Evaluation Committee, and editorial board of a number of prestigious international journals, including IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Cloud Computing, and IEEE Internet of Things Journal.

Junlong Zhou (Member, IEEE) is currently an Associate Professor at the Nanjing University of Science and Technology, Nanjing, China. His research areas are edge computing, cloud computing, and embedded systems, where he has published 110 refereed papers, including more than 40 in premier IEEE/ACM Transactions. He has held chair positions at many IEEE/ACM conferences. He has been an Associate Editor for Sustainable Computing: Informatics and Systems, Journal of Circuits, Systems, and Computers, and IET Cyber-Physical Systems: Theory & Applications, and a Subject Area Editor for the Journal of Systems Architecture. He received the Best Paper Awards from IEEE iThings 2020 and IEEE CPSCom 2022.