Regular Article

Fast Resource Allocation for Resilient Service Coordination in an NFV-Enabled Internet-of-Things System

Tuan-Minh Pham[1,2], **Thi-Minh Nguyen**[3]

[1] Faculty of Computer Science, Phenikaa University, Hanoi, Vietnam
[2] Phenikaa Research and Technology Institute (PRATI), A&A Phoenix Group JSC, Hanoi, Vietnam
[3] Department of Technology, Dong Nai Technology University, Dong Nai, Vietnam

Correspondence: Tuan-Minh Pham, minh.phamtuan@phenikaa-uni.edu.vn

Communication: received 12 July 2022, revised 20 December 2022, accepted 27 December 2022
Online publication: 03 January 2023, Digital Object Identifier: 10.21553/rev-jec.320
The associate editor coordinating the review of this article and recommending it for publication was Prof. Vo Nguyen Quoc Bao.

Abstract– **Network Functions Virtualization (NFV) is a new way of leveraging an Internet-of-Things (IoT) system to provide real-time and highly flexible service creation. In an NFV-enabled Internet-of-Things (NIoT) system, several IoT functions implemented as Virtual Network Functions can be linked as a service function chain to build a customized IoT service quickly. It is important for an IoT service to be able to recover from a failure. However, the supply of a resilient IoT service in an NIoT system is challenging due to the coordination of distributed VNF instances. In this paper, we formulate the problem of resilient service coordination in an NIoT system as a mixed-integer linear programming model, namely $\mathrm{RSO_d}$. The model offers the optimal resource allocation for minimizing service disruption when a failure happens at a node of an NIoT system. We also develop two modified versions of $\mathrm{RSO_d}$ for different use cases required by an IoT provider. Further, two approximation algorithms are proposed to provide a resilient service for a large-scale NIoT system. The evaluation results show that $\mathrm{RSO_d}$ and its modified versions produce the optimal resource allocation in significantly reduced time compared to previous work. The results suggest that an IoT provider should carefully select an appropriate resource allocation strategy as it has to pay a resource cost to minimize the service disruption. The results also show that our proposed priority-based heuristic algorithm outperforms an approximation algorithm based on Simulated Annealing in terms of the service disruption and computation time.**

Keywords– **NFV, NFV-enabled Internet-of-Things, NIoT, optimization, resilient service.**

1 Introduction

Many current Internet-of-Things (IoT) platforms rely on back-end Edge Cloud computing for virtually unlimited storage and processing capacity. However, the creation of a new IoT service is fairly static with respect to the dynamic association of IoT functions deployed distributedly in an Edge Cloud system. Network Function Virtualization (NFV) can overcome such a limitation by providing a framework in which Virtual Network Function (VNF) instances can be dynamically linked together as a service function chain (SFC) for the creation of flexible and real-time IoT services. An integrated NFV-IoT platform can be used to effectively operate many smart city applications, where the IoT functions are deployed as VNFs [1]. We will refer to such a virtualized IoT system based on NFV as NIoT.

While an NIoT system can offer the dynamic composition of an IoT service, it is challenging to provide a resilient IoT service due to the coordination of distributed VNF instances. The disruption of services can be caused by any system failure, such as a change in system configuration and security issues. A resilient NIoT system should rapidly reallocate IoT functions affected by a failure to maintain a service continuum. Previous studies have proposed several methods to address various problems of a robust service in NFV and IoT systems [2–5]. However, these solutions do not take into account service coordination, a crucial aspect of NFV in an NIoT system. Recently, Pham proposed an optimization model and approximation algorithms for service coordination based on SFC in NFV [6]. However, the issue of system failure has not been considered in the model. In addition, the proposed optimization model is limited to a small-scale problem. Our work is different as it finds an optimal resource allocation solution for IoT service coordination in an NIoT system under the presence of failures.

This paper is an extended version of our work presented at the 2022 international conference on Advanced Technologies for Communications (ATC) [7]. It extends the conference version paper by presenting new results of resource allocation for a resilient service in a large-scale NIoT system. Our main contributions are as follows. First, we propose a mixed-integer linear programming (MILP) model, namely $\mathrm{RSO_d}$, for the problem of resilient service coordination in an NIoT system. The $\mathrm{RSO_d}$ model provides the optimal resource allocation for minimizing the service disruption when a failure occurs in an NIoT system. The model has the advantage of being adaptable for addressing a variety of the resilient service coordination problem. Specifically, we develop two modified versions of RSO_d (i.e., RSO_r, RSO_m) for two use cases required by an IoT provider. Second, we propose an approximation algorithm based on Simulated Annealing and a heuristic algorithm to find an approximate solution in a scenario comprising hundreds of nodes and thousands of demands. Third, we validate our proposed models and algorithms in real and synthetic network topologies and provide suggestions for an IoT provider to choose an appropriate objective function. The evaluation results show that RSO_d, RSO_r, and RSO_m can provide the optimal resource allocation for a resilient service in an NIoT system. The results suggest that an IoT provider should carefully select an appropriate resource allocation strategy as it has to pay a resource cost to minimize the service disruption. In addition, the computation time of the proposed optimization models significantly improves in comparison to a previous optimization model of resilient services in NFV presented in [4]. The results also show that our proposed priority-based heuristic algorithm outperforms an approximation algorithm based on Simulated Annealing in terms of the service disruption and computation time.

The rest of the paper is organized as follows. Section 2 introduces some existing solutions for a resilient service in IoT and NIoT systems. Section 3 presents the system operation and the statement of the resource allocation optimization problem for resilient service coordination in an NIoT system. Section 4 presents MILP models that provide the optimal allocation of IoT functions for various use cases of resilient service coordination. Section 5 describes our proposed approximation solutions based on Simulated Annealing and heuristic approaches for a large scale IoT system. In Section 6, our proposed optimization models are evaluated based on three important performance measures, including a measure of service disruption, resource cost, and computation time. Finally, Section 7 concludes this paper with several potential research directions for future work.

2 Related Work

The use of NFV for IoT virtualization is an essential approach to processing a huge amount of data in an IoT system [8]. One of the most critical issues that has received much research attention is the resilience of services in an NFV-enabled IoT system. Many solutions have been proposed for service resilience in an IoT system [9–12]. For example, Abreu et al. designed a resilient IoT architecture, including IoT service, middleware, and infrastructure layers for smart cities [9]. However, the architecture does not support the connection between IoT services in different cloud systems. Ratasich et al. discussed the self-healing components in one node of a smart sensor infrastructure, which is not applicable to a chain of services [11]. In some research, the efficient design for data processing and routing in Internet of Vehicles (IoV) has been studied [13–15]. Si et al. proposed a resilient routing protocol to collect data from sensors in intra-car networks [13]. Malik et al. evaluated the performance of reliable packet dissemination protocols in vehicular ad hoc networks [14]. Muhammad et al. used mobile edge computing to improve the performance of data processing when the connections among vehicles are interrupted [15]. However, the coordination of distributed services has not been considered in an IoV design. Several optimization models for edge cloud computing's robust services have been proposed [16, 17]. Shahid et al. developed a load-balancing algorithm with fault tolerance in a cloud computing system [16]. Luo et al. discussed the collaboration for resource scheduling in edge computing [17]. However, the feature of service function chaining has not been considered in these proposals. In summary, these and other solutions of service resilience in a general IoT system, IoV or cloud computing cannot apply to an NIoT system because IoT functions in NIoT can be installed in various data centers and linked to one another to form a customized service.

Some studies have considered the resilient service problem in an NIoT system. Some approximate solutions have been proposed. For example, in [2], for the purpose of improving the robustness of NFV services implemented in a distributed edge network, the authors proposed a proactive fail-over technique based on failure prediction. In [3], the authors devised heuristics to boost an IoT network's fault tolerance when a system failure happens. In [5], the authors applied artificial intelligence approaches to give prediction for telemedicine applications. These solutions, however, do not take into account service function chaining, a crucial aspect of NFV. Recently, Pham proposed an optimization model and approximation algorithms for service function chaining in NFV [6]. However, the system failure issue has not been considered in the model. Our work is different as it determines an optimal resource allocation solution for the coordination of IoT services in an NIoT system while taking into account failures.

3 System Description

An NIoT system is an IoT virtualization framework based on NFV, including three components: NFV Infrastructure (NFVI), VNFs, and NFV Management and Orchestration (MANO) [1]. NFVI contains physical resources of NFV nodes, such as computation and storage, virtualized by cloud technologies. While a VNF instance in NFV is a traditional network function implemented as a software component packaged into a virtual machine image, a VNF instance in NIoT is an IoT function, e.g., smart metering, access control, and data compression. The MANO component covers the lifecycle management of physical resources and VNFs. A resource allocation strategy for service resilience can be deployed as a component of MANO.

In an NIoT system, IoT functions can be coordinated in a service function chain to build an IoT service. For example, an SFC of a video surveillance service is composed of three IoT functions, including video compression, access control, and video decompression. MANO allocates NFVI resources for an IoT service requested by a user. When a system failure such as a configuration change or security issue happened at an NFV node, MANO should quickly find a new location of an IoT function affected by the failure.

The NIoT system $G = (V, E)$ contains physical nodes V and links E. Each node $v \in V$ has a compute capacity r^n_v, i.e., the number of CPU cores. Each link $e \in E$ associated with the beginning node i_e and the ending node j_e has a bandwidth capacity r^l_e. The system supports a set of VNF types F. A VNF type $u \in F$ requires some cores η_u to process a unit of traffic volume. The amount of time that node v needs to route a unit of traffic volume is the routing delay δ_{2v}. The amount of time needed by node v to process VNF type u is the processing delay μ_{vu}. λ^n_v are the failure state of node v. $\lambda^n_v \in [0,1]$ represents the percentage of node v 's resource remaining after a failure happened at v. $\lambda^n_v = 0$ if node v completely fails.

The NIoT system provides a set of SFCs $\Omega = \{S_i = (u_{i1}, \ldots, u_{ij}, \ldots, u_{in})\}$ where u_{ij} is the jth VNF of SFC S_i. We denote the service demand set by $\Gamma = \{d\}$. A service demand $d \in \Gamma$ is defined by source node s_d, destination node t_d, SFC $S_d \in \Omega$, SFC delay δ_{1d}, and bandwidth volume b_d.

A NIoT system must find a new resource allocation solution for IoT functions after a failure in order to meet the demand requirements. We assume that the NIoT system employs one of the popular intra-routing protocols, such as Open Shortest Path First (OSPF), which uses the shortest path first algorithm for path selection. MANO optimizes the new location of IoT functions along the shortest path. The performance of an NIoT system may be dramatically impacted by a reallocation strategy, including the placement and coordination of IoT functions. The problem of resource allocation for resilient service coordination is stated as follows:

Problem 1 (Resource Allocation Optimization for Resilient Service Coordination (RS)). *Given an NIoT system G, find a resource allocation solution for a set of service demands Γ, in order to minimize the system disruption when a failure occurs under constraints on the availability of system resource and the coordination of IoT functions.*

4 Optimization Model for Resilient Service Coordination

We propose an MILP model (i.e., RSO_d) to obtain the optimal resource allocation for the RS problem. We represent the solution by the following variables:

- $\mathbf{l}_2 = \left(l_{2ep_{s_dt_d}}\right)$ is the new link selection for traffic routing when a failure occurs. If path $p_{s_dt_d}$ uses link e, $l_{2ep_{s_dt_d}} = 1$, otherwise, $l_{2ep_{s_dt_d}} = 0$.
- $\mathbf{h}_2 = \left(h_{2vp_{s_dt_d}}\right)$ is the new node selection for traffic routing when a failure occurs. If path $p_{s_dt_d}$ uses node v, $h_{2vp_{s_dt_d}} = 1$, otherwise, $h_{2vp_{s_dt_d}} = 0$.
- $\mathbf{y}_2 = (y_{2vdi})$ is the new IoT function placement when a failure happens. If the ith function of demand d is supplied by node v, $y_{2vdi} = 1$, otherwise, $y_{2vdi} = 0$.

Table I provides a summary of RSO_d's notations.

4.1 Resource Allocation for Service Demand

The routing decision variables $\mathbf{l}_2$ and $\mathbf{h}_2$ are determined before we find a placement solution as we assume that an NIoT system uses the shortest path first algorithm for path selection. The service placement condition is as follows:

$$\textstyle\sum_v y_{2vdi} = 1, \quad \forall d, \forall i, \tag{1}$$

$$y_{2vdi} \leqslant h_{2vdi}, \quad \forall v, \forall d, \forall i. \tag{2}$$

Formula (1) guarantees that the ith IoT function of demand d is allocated by one node in the system. Formula (2) assures that the ith IoT function of demand d is supplied by node v only if the path of demand d contains v.

The system loses a part of the compute capacity when a node fails. The condition of the available compute capacity is as follows:

$$\textstyle\sum_{d,i} b_d y_{2vdi} \eta_{u_{di}} \leqslant r^n_v - r^n_v \lambda^n_v, \quad \forall v. \tag{3}$$

Formula (3) ensures that the total number of cores allocated by node v to all IoT functions of all demands is less than or equal to the remaining amount of cores of node v after a failure.

The delay of a service demand includes the routing and function processing delay. The condition of a delay guarantee is as follows:

$$\textstyle\sum_v \delta_{2v} h_{2vp_{s_dt_d}} b_d + \sum_v \mu_{vu_{di}} \sum_i y_{2vdi} \leqslant \delta_{1d}, \quad \forall d. \tag{4}$$

Formula (4) ensures that the sum of the total routing delay along demand d's path and the total function processing delay of all IoT functions required by demand d is less than or equal to the delay requirement of demand d.

4.2 Service Coordination

We use two extra variables y^σ_{2vdi} and $\bar{y}^\sigma_{2edi}$ to express the condition of service coordination. If a node between s_d and v on demand d's path allocates function u_{di}, $y^\sigma_{2vdi} = 1$, otherwise $y^\sigma_{2vdi} = 0$. If link e belongs to demand d's path, and a node between s_d and i_e allocates function u_{di}, $\bar{y}^\sigma_{2edi} = 1$, otherwise $\bar{y}^\sigma_{2edi} = 0$. The condition of service coordination is as follows:

$$y_{2vdi} \leqslant y^\sigma_{2vd(i-1)}, \quad \forall v, \forall d, \forall i \geqslant 1, \tag{5}$$

$$y^\sigma_{2vdi} = y_{2vdi} + \textstyle\sum_{\{e:j_e=v\}} \bar{y}^\sigma_{2edi}, \quad \forall v, \forall d, \forall i, \tag{6}$$

$$l_{2p_{s_dt_d}e} + y^\sigma_{2i_edi} - 1 \leqslant \bar{y}^\sigma_{2edi} \leqslant l_{2p_{s_dt_d}e}, \quad \forall e, \forall d, \forall i, \tag{7}$$

$$\bar{y}^\sigma_{2edi} \leqslant y^\sigma_{2i_edi}, \quad \forall e, \forall d, \forall i. \tag{8}$$

Formula (5) ensures that function u_{di} can be allocated by node v if and only if function $u_{d(i-1)}$ is provided by either v or its previous node along demand d's path.

Table I
Summary of Main Notations

Input Parameters	
$G = (V, E)$	The NIoT system contains physical node V and links E.
r^n_v	Node v's compute capacity
r^l_e	Link e's bandwidth capacity
i_e	Link e's beginning node
j_e	Link e's ending node
F	The set of IoT function types
η_u	The number of CPU cores necessary for processing a unit of flow data using function type $u \in F$
Ω	The set of SFCs $\Omega = \left\{S_i = \left(u_{i1}, \ldots, u_{ij}, \ldots, u_{in}\right)\right\}$ where u_{ij} is the jth IoT function of SFC S_i
$\Gamma = \{d\}$	The service demand set
s_d	Demand d's source node
t_d	Demand d's destination node
b_d	Demand d's bandwidth volume
S_d	Demand d's SFC
δ_{1d}	Demand d's SFC delay
δ_{2v}	Node v's routing delay for a traffic unit
μ_{vu}	Node v's processing delay of function type u
$\gamma_{vv'u_{di}}$	The time amount necessary to migrate demand d's ith function from v to v'
$\rho_{vv'}$	The time amount required to transfer a data volume on the minimum-weight path from v to v'
κ_u	The data volume of function type u
λ^n_v	The percentage of node v's resource remaining in the failure state
$\mathbf{l}_1 = \left(l_{1ep_{s_dt_d}}\right)$	The link selection in the current routing solution: If path $p_{s_dt_d}$ uses link e, $l_{1ep_{s_dt_d}} = 1$, otherwise, $l_{1ep_{s_dt_d}} = 0$
$\mathbf{h}_1 = \left(h_{1vp_{s_dt_d}}\right)$	The node selection in the current routing solution: If path $p_{s_dt_d}$ uses node v, $h_{1vp_{s_dt_d}} = 1$, otherwise, $h_{1vp_{s_dt_d}} = 0$
$\mathbf{y}_1 = (y_{1vdi})$	The current placement of IoT functions when a failure happens. If node v provides the ith function of demand d, $y_{1vdi} = 1$, otherwise, $y_{1vdi} = 0$.
Output variables	
$\mathbf{l}_2 = \left(l_{2ep_{s_dt_d}}\right)$	The link selection in the routing solution when a failure occurs: If path $p_{s_dt_d}$ uses link e, $l_{2ep_{s_dt_d}} = 1$, otherwise, $l_{2ep_{s_dt_d}} = 0$
$\mathbf{h}_2 = \left(h_{2vp_{s_dt_d}}\right)$	The node selection in the routing solution when a failure occurs: If path $p_{s_dt_d}$ uses node v, $h_{2vp_{s_dt_d}} = 1$, otherwise, $h_{2vp_{s_dt_d}} = 0$
$\mathbf{y}_2 = (y_{2vdi})$	The new placement of IoT functions when a failure happens. If the ith function of demand d is supplied by node v, $y_{2vdi} = 1$, otherwise, $y_{2vdi} = 0$.

Formula (6) assures that $y^{\sigma}_{2vdi} = 1$ if and only if function u_{di} is allocated by v or its previous node along demand d's path. Formulas (7) and (8) ensures that $\bar{y}^{\sigma}_{2edi} = 1$ if and only if function u_{di} is allocated by either i_e or its preceding node, and demand d path contains link e.

4.3 Minimizing Service Disruption

The measure of service disruption is defined as a derived unit of time required to restore all service demands after a failure. We denote by $\gamma_{vv'u_{di}}$ the time amount required to migrate demand d's ith function from v to v'. $\rho_{vv'}$ is the time amount required to transfer a data volume on the shortest path from v to v'. κ_u is the data volume of function type u. The service disruption measure of an IoT function moved from v to v' is given by:

$$\gamma_{vv'u_{di}} = \rho_{vv'}\kappa_{u_{di}}. \tag{9}$$

An extra variable $z_{vv'di}$ is used to compute the service disruption measure of the NIoT system. After a failure, if demand d's ith function is changed from v to v', $z_{vv'di} = 1$, otherwise, $z_{vv'di} = 0$. We denote by $\mathbf{y}_1 = (y_{1vdi})$ the current IoT function placement. If node v allocates demand d's ith function, $y_{1vdi} = 1$, otherwise, $y_{1vdi} = 0$. The compatibility of $z_{vv'di}$ is given by:

$$z_{vv'di} \leqslant y_{1vdi}, \quad \forall v, \forall v', \forall d, \forall i, \tag{10}$$

$$z_{vv'di} \leqslant y_{2v'di}, \quad \forall v, \forall v', \forall d, \forall i, \tag{11}$$

$$y_{1vdi} + y_{2v'di} - 1 \leqslant z_{vv'di}, \quad \forall v, \forall v', \forall d, \forall i. \tag{12}$$

Formulas (10)–(12) assure that $z_{vv'di} = 1$ if and only if $y_{1vdi} = 1$ in the current solution of IoT function placement and $y_{2v'di} = 1$ in the new solution of IoT function placement, otherwise, $z_{vv'di} = 0$.

The service disruption measure of the NIoT system after a failure is as follows:

$$U_d = \textstyle\sum_{v,v',d,i} z_{vv'di}\gamma_{vv'u_{di}}. \tag{13}$$

The RSO_d model include the objective function defined by formula (13) and the constrains described by formulas (1)–(12). The RSO_d model provides the optimal IoT function placement for service demands after the routing path selection, which is determined by a common routing protocol.

4.4 Extensions of $\mathrm{RSO_d}$

We can modify $\mathrm{RSO_d}$ to find the optimal resource allocation to service demand for different requirements of IoT providers. We model the two following use cases as examples of $\mathrm{RSO_d}$ extensions. In the first use case, we consider that an IoT provider can focus on minimizing resource usage during pick load. Let c_v be the cost for providing one core at node v. The objective function that computes the resource cost is as follows:

$$U_r = \sum\nolimits_{d,v,i} b_d y_{2vdi} \eta_{u_{di}} c_v. \tag{14}$$

We name the optimization model that minimizes the resource usage as $\mathrm{RSO_r}$. Formula (14) describes the objective function of $\mathrm{RSO_r}$. Formulas (1)–(12) are the constrains of $\mathrm{RSO_r}$.

In the second use case, an IoT provider wants to select the solution with the lowest resource usage when there exist many optimal solutions of VNF allocation that minimize the service disruption measure. We denote by α a scale parameter. The objective function is modified as follows:

$$U_m = U_d + \alpha U_r. \tag{15}$$

Cococcioni et al. presented a principle of selecting a value for α [18]. The general idea is to multiply the most important objective by one and the second by a small number. We suggest $\alpha = 1/\max U_r$. By multiplying U_r and α in the objective function, we prioritize minimizing the service disruption measure more than reducing the resource usage. We can select a different value of α to obtain a trade-off between the priority of the service disruption measure and that of the resource usage.

We name the optimization model that takes into account the optimization of both the service disruption measure and resource usage as $\mathrm{RSO_m}$. The objective function and the constraints of $\mathrm{RSO_m}$ are formula (15) and formulas (1)–(12), respectively. The above discussions demonstrate that $\mathrm{RSO_d}$ can be adaptable for different use cases required by an IoT provider.

5 Approximate Solutions to Resource Allocation for Resilient Service Coordination

The there proposed models based on MILP (i.e., $\mathrm{RSO_d}$, $\mathrm{RSO_r}$, $\mathrm{RSO_m}$) provide the optimal resource allocation for resilient service coordination in an NIoT system after a failure. However, we cannot solve the optimization models in a large-scale NIoT system. In this section, we develop two algorithms based on approximation and heuristic approaches to find an approximate solution to the RS problem in a large-scale NIoT system.

5.1 Simulated Annealing-based Approximation Algorithm for RS

We propose an approximation algorithm, namely RSS, based on the Simulated Annealing (SA) [19]. In RSS, we develop the representation of the resource allocation solution and the neighborhood selection function for the RS problem.

A resource allocation solution for the RS problem $O_m = ((d,i,v) : d \in D, i \in S_d, v \in V)$ represent that node v provides the ith function of demand d. We add a dummy function to the source and destination nodes of a service demand for presenting the neighborhood selection. The dummy function is an IoT function that does not require any resources. The SFC of demand d is modified as follows: $S_d = \left(u_{d0}, u_{d1}, \ldots, u_{ij}, \ldots, u_{in}, u_{i(n+1)}\right)$ where u_{d0} and $u_{d(n+1)}$ are the dummy functions and n is the original number of IoT functions requested by demand d. We describe all steps of RSS in Algorithm 1.

Algorithm 1: Simulated Annealing-based approximation algorithm for RS

```
1:  function RSS($G$, $\Gamma$, $\mathbf{y}_1$)
2:      Initialize $T$, $T_0$, $T_n$, $\phi$, $\tau$
3:      $V_d \leftarrow$ a sequence of nodes on the demand path
        from $s_d$ to $t_d$
4:      Find an initial solution $O_m$
5:      $O_m^* \leftarrow O_m$
6:      Compute $\mathbf{y}_2^*$ from $O_m^*$
7:      while $T \geq T_n$ do
8:          for $n \leftarrow 1$ to $\phi$ do
9:              repeat    ▷ Find a neighborhood solution
10:                 $(d,i,v) \leftarrow$ a random tuple in $O_m$
11:                 $j_s \leftarrow$ index of $v$ in $V_d$ where
                    $y_{2vd(i-1)} = 1$
12:                 $j_t \leftarrow$ index of $v$ in $V_d$ where $y_{2vd(i+1)}=1$
13:                 $j \leftarrow$ a random node in $[j_s, j_t]$
14:                 $v' \leftarrow V_d(j)$
15:                 $O_m' \leftarrow \textsc{Replace}(d,i,v,v',O_m)$
16:             until $O_m'$ is feasible
17:             Compute $\mathbf{y}_2$ from $O_m$
18:             Compute $\mathbf{y}_2'$ from $O_m'$
19:             if $U_d(\mathbf{y}_2',\mathbf{y}_1) < U_d(\mathbf{y}_2,\mathbf{y}_1)$ then    ▷ The
                neighborhood solution is better
20:                 $O_m \leftarrow O_m'$
21:                 if $U_d(\mathbf{y}_2',\mathbf{y}_1) < U_d(\mathbf{y}_2^*,\mathbf{y}_1)$ then
22:                     $O_m^* \leftarrow O_m'$
23:                     $\mathbf{y}_2^* \leftarrow \mathbf{y}_2'$
24:                 end if
25:             else    ▷ The current solution is better
26:                 $\Delta \leftarrow U_d(\mathbf{y}_2',\mathbf{y}_1) - U_d(\mathbf{y}_2,\mathbf{y}_1)$
27:                 $\varepsilon \leftarrow$ a random number in the range 0
                    to 1
28:                 if $\exp(-\Delta/T) > \varepsilon$ then
29:                     $O_m \leftarrow O_m'$    ▷ Move to the
            neighborhood solution with a probability
30:                 end if
31:             end if
32:         end for
33:         $T \leftarrow C(T)$
34:     end while
35:     return $\mathbf{y}_2^*$, $\mathbf{x}_2^*$
36: end function
```

RSS includes the While loop (i.e., line 7) and the For loop (i.e., line 8). RSS stops when the value of the temperature parameter T is less than that of the stop temperature parameter T_n. T_n is close to zero. The initial value of T is defined by T_0, which can be the maximal value difference of the objective function. T decreases by the cooling function $C(T)$ after one iteration of the While loop. A simple cooling function is $C(T) = \tau T$ for $\tau \in (0, 1)$.

RSS runs the For loop for each T. The number of iterations of the For loop is denoted by ϕ. RSS repeatedly changes the current solution to another feasible solution in the loop by a neighborhood selection procedure. Let O_m be an initial solution. In the neighborhood selection (i.e., line 9–16), we first randomly select the placement of an IoT function of a demand in the current solution (i.e., $(d, i, v) \in O_m$). We then build a new solution by replacing node v with a feasible node v' (i.e., Replace(d, i, v, v', O_m)). The neighborhood selection process is repeated to identify a new feasible solution (i.e., line 16).

After RSS obtains a new feasible solution, RSS replaces the current solution with the neighborhood solution if the neighborhood solution is better than the current solution (i.e., lines 19–24). Otherwise, to overcome local optimization, RSS probabilistically switches the current solution to the neighborhood solution (i.e., lines 25–30). The computation time and approximation of RSS's solution are controlled by ϕ and $C(T)$.

5.2 Priority-based Heuristic Algorithm for RS

We propose a priority-based heuristic algorithm, namely RSP, to find an approximate solution to resource allocation for resilient service coordination in a large-scale NIoT system. The main idea of RSP is based on some priority indexes of demand and node selection. We describe all steps of RSP in Algorithm 2.

For the demand selection process (i.e., lines 2–5), RSP only considers the list of demands whose path is affected by a failure. RSP sorts the list in descending order of the total computing resources necessary to provide all functions of a demand. We argue that the number of feasible resource allocation solutions to a demand increases due to the reduction of the fragmentation of computing resources stored in separate nodes when we process a demand that requires a large number of computing resources first. Hence, RSP might find a good approximate solution. RSP then processes all demands one by one in the list.

For the node selection process of each demand d (i.e., lines 6–18), RSP begins by finding the first node on path $p_{s_d t_d}$ that can provide an IoT function required by d. It then determines the best node defined by index function $\Psi(v)$ for an IoT function in the reverse sequence in SFC (i.e., lines 14–16). Let η_v be the number of $p_{s_d t_d}$ that contains v for all $d \in \Gamma_f$. The index function of node selection for an IoT function is defined as follows:

$$\Psi(v) = \frac{r_v^n}{\eta_v}. \tag{16}$$

Algorithm 2: Priority-based heuristic algorithm for RS

```
 1: function RSP($G$, $\Gamma$, $\mathbf{y}_1$)
 2:     $\Gamma_f \leftarrow$ The list of demands whose path is affected
        by a failure
 3:     Sort $\Gamma_f$ in descending order of the total number
        of cores necessary to provide all functions of a
        demand
 4:     Find $p_{s_d t_d}$ for all $d \in \Gamma_f$
 5:     for all $d \in \Gamma_f$ do
 6:         $\Omega_d \leftarrow \emptyset$          ▷ Initialize the list of nodes
            providing $S_d$
 7:         for all $u_{di} \in S_d$, $i = 1 \dots |S_d|$ do
 8:             $v_i \leftarrow$ the first node on path $p_{s_d t_d}$ that can
                provide $u_{di}$
 9:             $\Omega_d \leftarrow \Omega_d \cup \{v_i\}$
10:         end for
11:         Return failure if solution not found
12:         $v_{|S_d|+1} \leftarrow t_d$
13:         for all $u_{di} \in S_d$, $i = |S_d| \dots 1$ do
14:             $P \leftarrow$ the list of nodes that can provide $u_{di}$
                along path $p_{s_d t_d}$ from $v_i$ to $v_{i+1}$
15:             Find $v_* \leftarrow \arg\max_{v \in P} \Psi(v)$ ▷ Use the index
                function to find the best node
16:             Replace $v_i \in \Omega_d$ with $v_*$ if node found
17:             Update the compute capacity of $v_i$ and $v_*$
18:         end for
19:     end for
20:     return $\Omega_d$
21: end function
```

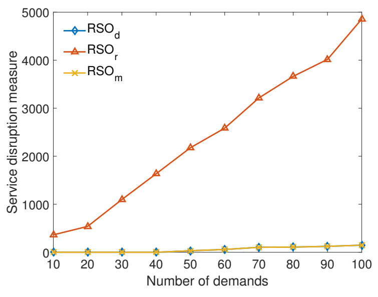
Using the index function, we select a node whose remaining resources are maximized for an IoT function when considering the possibility of resource sharing in demand paths containing a similar node.

6 Evaluation

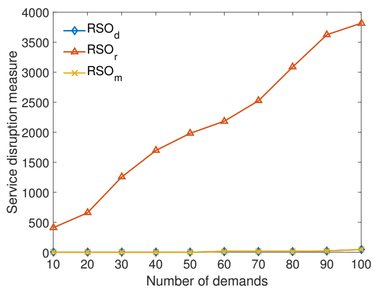
In this section, we evaluate our proposed optimization models and approximation solutions of resource allocation for resilient service coordination in an NIoT system. We first present the scenarios and parameters in our evaluation using synthetic topologies and real-world datasets. We then analyze the model performance in terms of the service disruption measure, resource cost, and computation time. We use the optimal solution obtained by RSO as a baseline solution for analyzing the performance of RSS and RSP. We also discuss a suggestion to an IoT provider for efficiently supplying users with a resilient service.

6.1 Scenarios and Parameters Setting

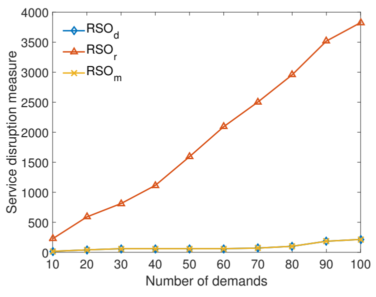
Our evaluation scenarios include the Geant, Barabási-Albert (BA), Waxman (WA) topologies. The Geant topology describes the 2012 Europe backbone network, composed of 40 nodes and 61 links [20]. The BA and WA topologies are generated by the Barabási-Albert and Waxman models, respectively [21]. They consist of 50 nodes. In the BA topology, we initially generate four



(a) Geant

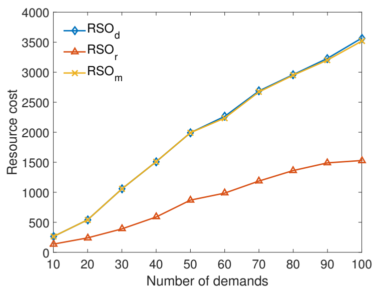


(b) BA

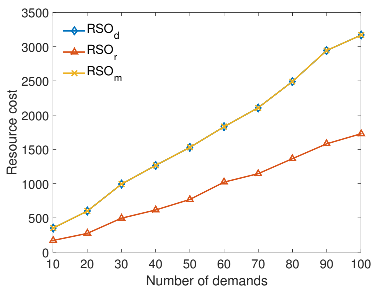


(c) WA

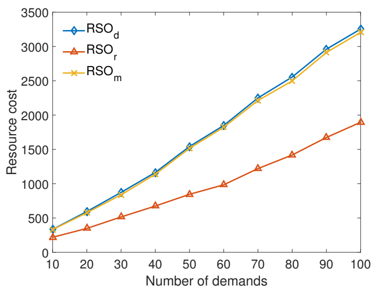
Figure 1. Service disruption.



(a) Geant



(b) BA



(c) WA

Figure 2. Resource cost.

nodes. We then repeatedly add a new node and connect it to four existing nodes. The WA topology is built using a link density probability of 0.9. We build a network topology with a node fault rate of 20%. The percentage of node v's resource remaining after a failure is chosen randomly between 0 and 1.

The source and destination nodes are chosen randomly in each service demand. There is a range of one to thirty milliseconds for the SFC delay. The bandwidth requirement ranges from 1 Gbps to 5 Gbps. Four types of IoT functions are implemented as VNFs in an evaluation scenario. A VNF type may require between one and two CPU cores to handle a unit of traffic volume. The ordered list of IoT functions is chosen at random in each service demand. The number of cores available per node is 200. We give each link's capability a bandwidth value of 80 Gbps. The link weight value ranges from 1 to 3. The processing delay of an IoT function and the routing delay for a traffic unit are created randomly and range from 10 to 100 microseconds at a node.

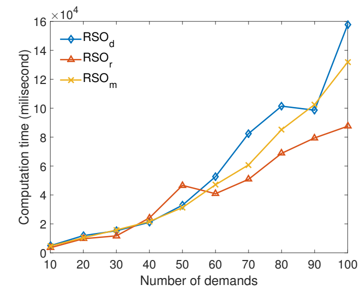
We implemented RSO$_d$, RSO$_r$, RSO$_m$ in CPLEX [22]. We used an x86 with a four-core 2.60 GHz Intel processor and 8 GB memory to solve these models when varying the number of service demands between 10 and 100 demands.

6.2 The Performance of Optimization Models

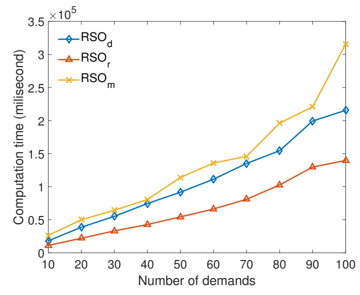
First, we compute the service disruption measure provided by the three optimization models in the Geant, BA, and WA topologies. Figure 1 show that the service disruption measure of RSO$_d$ and RSO$_m$ is better than that of RSO$_r$. This was to be expected since the objective function of RSO$_r$ does not include the service disruption measure. RSO$_d$ and RSO$_m$ obtain the similar service disruption measure because RSO$_m$ gives a high priority to the service disruption measure in its objective function. We also observe that the service disruption measure of RSO$_d$ in BA is slightly less than that in WA. This occurs because the BA network is quite dense (i.e., the number of links in BA is 253 and the number of links in WA is 98), letting an MILP model have better alternative solutions to optimize through the solution space.

Second, we compare the resource cost of different optimization models. Figure 2 shows that RSO$_r$ is better than other models and RSO$_d$ is the worst in term of the resource cost. On the contrary, Figure 1 shows that RSO$_d$ is better than other models in terms of the service disruption measure. It refers that an IoT provider pays some resource cost to minimize the service disruption measure. RSO$_m$ is slightly better than RSO$_d$ in the BA topology because RSO$_m$ takes into account both the service disruption measure and resource cost while RSO$_d$ only consider the service disruption measure. The results suggest that an IoT provider's possible resource allocation strategy is to select an appropriate value of α for a tradeoff between the service disruption measure and resource cost.

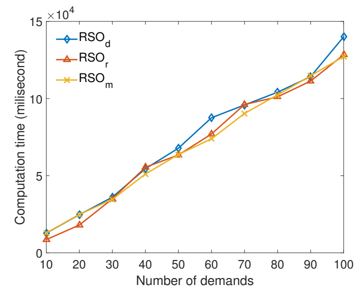
Finally, we evaluate the computation time of our proposed optimization models. Figure 3 plots the computation time in the Geant, BA, and WA topologies when varying the number of service demands between 10 and 100. In such topologies, the PT-O optimization model for a resilient NFV service presented in [6] was unable to provide an optimal resource allocation. PT-O can produce the optimal result for a scenario of 22 nodes, 59 links, and 64 demands. Our proposed models can obtain the optimal solution in an acceptable time



(a) Geant

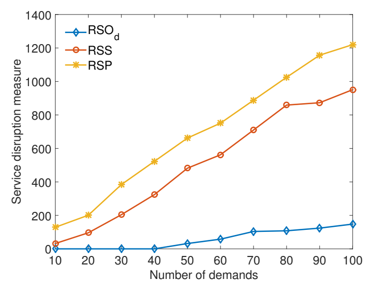


(b) BA

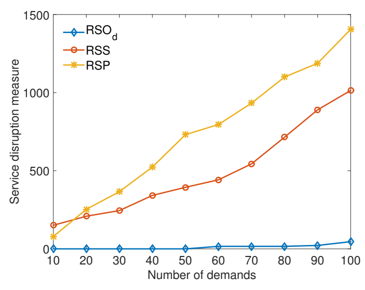


(c) WA

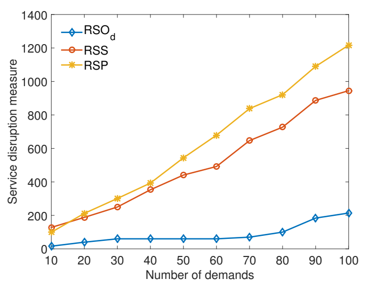
Figure 3. Computation time.



(a) Geant

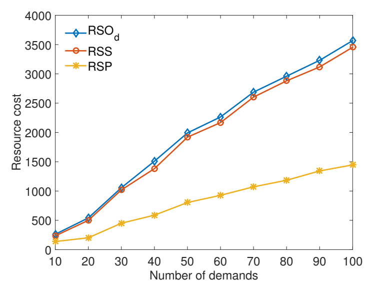


(b) BA

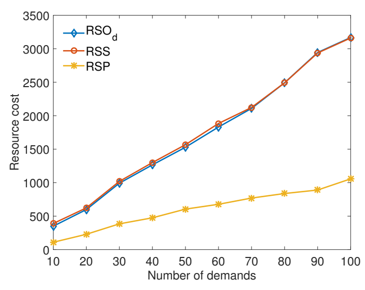


(c) WA

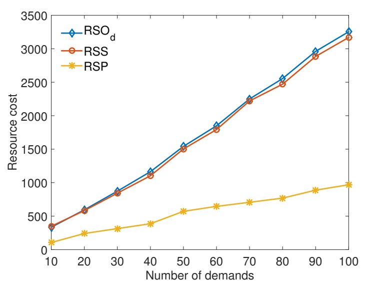
Figure 4. Comparison between the approximate solution with the optimal solution in terms of the service disruption measure.



(a) Geant

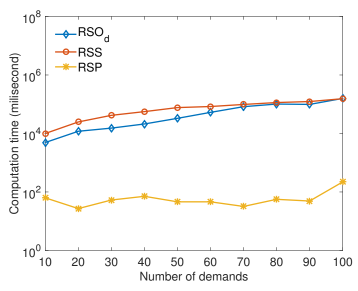


(b) BA

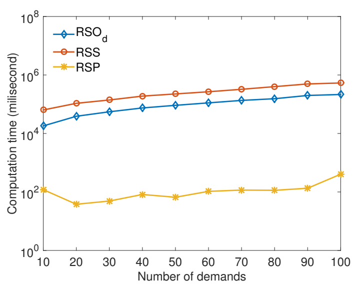


(c) WA

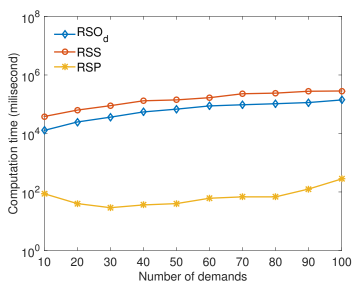
Figure 5. Comparison between the approximate solution with the optimal solution in terms of the resource cost.



(a) Geant



(b) BA



(c) WA

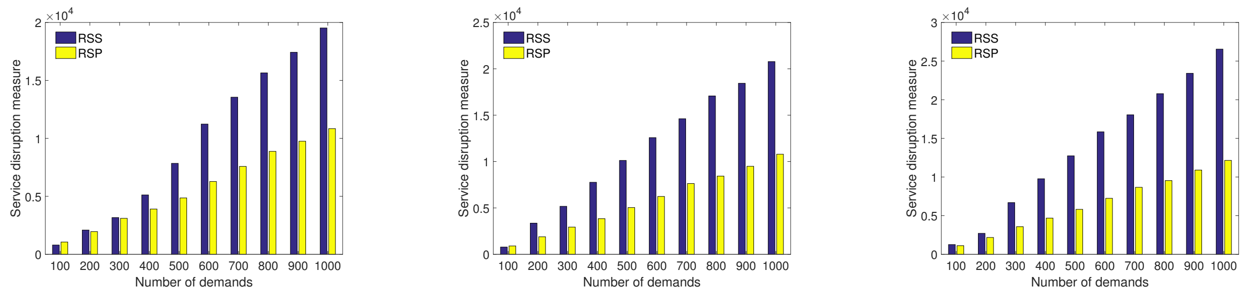
Figure 6. Comparison between the approximate solution with the optimal solution in terms of the computation time.

for a scenario of 50 nodes, 353 links, and 100 demands, which is a significant improvement.

6.3 The Performance of RSS and RSP

We first compare approximate solutions produced by RSS and RSP with the optimal solution obtained by RSO_d as we use U_d as the objective function in RSS and RSP. We compute the service disruption measure, resource cost, and computation time of RSS and RSP when varying the number of demands between 10 and 100. To depict the difference in the computation time, we use a base-ten logarithmic scale for the y-axis in Figure 6. The results presented in Figures 4, 5, and 6 show that RSS can provide a better approximate solution compared to RSP in the service disruption measure. However, it is very time-consuming for RSS to obtain its solution.

We then evaluate the performance of RSS and RSP



(a) Geant (b) BA (c) WA

Figure 7. Comparison between RSS and RSP in terms of the service disruption measure in a large-scale scenario.

in a large-scale NIoT system. The number of demands is between 100 and 1000. In all scenarios, it takes less than 3 seconds for RSP to find its solution to the RS problem. We collect the results obtained by RSS after 10 minutes even though it does not finish for all scenarios in which the number of demands is more than 200. Figure 7 shows that RSP outperforms RSS in the service disruption measure when we restrict the running time of algorithms in a large-scale NIoT system. Hence, when it is required to obtain an approximate solution in a limited time, an IoT provider should consider RSP as its resource allocation strategy for resilient service coordination.

7 Conclusions

We addressed the optimization problem of resource allocation for resilient service coordination in an NIoT system. We developed an MILP model (i.e., $\mathrm{RSO_d}$) that finds the optimal resource allocation to minimize the service disruption after a failure. The model takes into account an ordered list of IoT functions in a service request, the shortest path routing, and the reallocation of IoT functions in an NIoT system when a failure occurs. We discussed two extended versions of the model, including $\mathrm{RSO_r}$ and $\mathrm{RSO_m}$ for two different use cases required by an IoT provider. For a large-scale NIoT system, we developed the RSS algorithm based on Simulated Annealing and the RSP algorithm based on a heuristic approach to find an approximate solution. The evaluation results show that $\mathrm{RSO_d}$, $\mathrm{RSO_r}$, and $\mathrm{RSO_m}$ can archive the optimal resource allocation solution for resilient services in an NIoT system after a failure. They suggest that an IoT provider should carefully select an appropriate resource allocation strategy to obtain a tradeoff between the optimization of service disruption measure and that of the resource cost. In addition, our proposed optimization models significantly outperform a previous optimization model in the computation time. The results also show that RSP is better than RSS in terms of the service disruption and computation time when the number of service demands is large. Our future work will address the dynamics of service demand parameters and various performance metrics as in [6, 23]. We also can consider a more general coordination model or the federation of IoT providers for enhancing service quality.

Author Contributions

Problem definition, T.-M.P.; optimization models, T.-M.P.; software, T.-M.N; validation, T.-M.N. and T.-M.P.; writing–original draft preparation, T.-M.P.; writing–review and editing, T.-M.P. and T.-M.N. All authors have read and agreed to the published version of the manuscript.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.02-2020.13.

References

[1] ETSI, "SmartM2M: virtualized IoT architectures with cloud back-ends, TR 103 527 V1.1.1," 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_tr/103500_103599/103527/01.01.01_60/tr_103527v010101p.pdf

[2] H. Huang and S. Guo, "Proactive failure recovery for NFV in distributed edge computing," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 131–137, 2019.

[3] D. Ergenc, J. Rak, and M. Fischer, "Service-based resilience for embedded IoT networks," in *Proceedings of the 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2020, pp. 540–551.

[4] T.-M. Pham, S. Fdida, T.-T.-L. Nguyen, and H.-N. Chu, "Modeling and analysis of robust service composition for network functions virtualization," *Computer Networks*, vol. 166, p. 106989, 2020.

[5] L. Sanabria-Russo, J. Serra, D. Pubill, and C. Verikoukis, "CURATE: on-demand orchestration of services for health emergencies prediction and mitigation," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 2, pp. 438–445, 2021.

[6] T.-M. Pham, "Optimizing service function chaining migration with explicit dynamic path," *IEEE Access*, vol. 10, pp. 16 992–17 002, 2022.

[7] T.-M. Pham, T.-M. Nguyen, X.-T.-T. Nguyen, H.-N. Chu, and N. H. Son, "Fast optimal resource allocation for resilient service coordination in an NFV- enabled internet-of-things system," in *Proceedings of the International Conference on Advanced Technologies for Communications (ATC)*, 2022, pp. 141–146.

[8] T.-M. Pham and T.-T.-L. Nguyen, "Optimization of resource management for NFV-enabled IoT systems in edge cloud computing," *IEEE Access*, vol. 8, pp. 178 217–178 229, 2020.

[9] D. P. Abreu, K. Velasquez, M. Curado, and E. Monteiro, "A resilient internet of things architecture for smart

cities," *Annals of Telecommunications*, vol. 72, no. 1, pp. 19–30, 2017.

[10] O. Kaiwartya, A. H. Abdullah, Y. Cao, J. Lloret, S. Kumar, R. R. Shah, M. Prasad, and S. Prakash, "Virtualization in wireless sensor networks: Fault tolerant embedding for internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 571–580, 2018.

[11] D. Ratasich, F. Khalid, F. Geissler, R. Grosu, M. Shafique, and E. Bartocci, "A roadmap toward the resilient internet of things for cyber-physical systems," *IEEE Access*, vol. 7, pp. 13 260–13 283, 2019.

[12] C. Berger, P. Eichhammer, H. P. Reiser, J. Domaschka, F. J. Hauck, and G. Habiger, "A survey on resilience in the IoT: taxonomy, classification, and discussion of resilience mechanisms," *ACM Computing Surveys*, vol. 54, no. 7, sep 2021.

[13] W. Si, D. Starobinski, and M. Laifenfeld, "A robust load balancing and routing protocol for intra-car hybrid wired/wireless networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, p. 250–263, 2019.

[14] F. M. Malik, H. A. Khattak, A. Almogren, O. Bouachir, I. U. Din, and A. Altameem, "Performance evaluation of data dissemination protocols for connected autonomous vehicles," *IEEE Access*, vol. 8, pp. 126 896–126 906, 2020.

[15] A. Muhammad, M. Saqib, and W.-C. Song, "Sensor virtualization and data orchestration in internet of vehicles (IoV)," in *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 998–1003.

[16] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, and S. Musa, "A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach," *IEEE Access*, vol. 8, pp. 130 500–130 526, 2020.

[17] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: A survey," *IEEE Communications Surveys Tutorials*, vol. 23, no. 4, pp. 2131–2165, 2021.

[18] M. Cococcioni, M. Pappalardo, and Y. D. Sergeyev, "Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm," *Applied Mathematics and Computation*, vol. 318, pp. 298–311, 2018, recent Trends in Numerical Computations: Theory and Algorithms.

[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

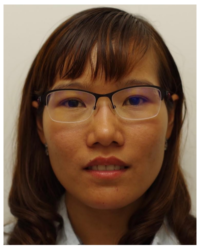
[20] "Geant dataset." [Online]. Available: http://www.topology-zoo.org/dataset.html

[21] M. Drobyshevskiy and D. Turdakov, "Random graph modeling: A survey of the concepts," *ACM Computing Surveys*, vol. 52, no. 6, 2019.

[22] "IBM ILOG CPLEX Optimizer." [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer/

[23] T.-M. Pham, "Traffic engineering based on reinforcement learning for service function chaining with delay guarantee," *IEEE Access*, vol. 9, pp. 121 583–121 592, 2021.

Tuan-Minh Pham (Senior Member, IEEE) received the Ph.D. degree in computer science from the University Pierre et Marie Curie, France, in 2011. He was a Visiting Scientist with Pennsylvania State University and the University Pierre et Marie Curie, in 2013 and 2017, respectively. He currently holds a faculty position at the Department of Computer Science, Phenikaa University. His research interests include the future Internet architectures, the modeling and analysis of networked systems, and network measurement for protocol evaluation. He was a recipient of the Best Paper Award from the IEEE International Conference on Social Computing in 2013 and the Elsevier Computer Networks in 2021. He has served as a Technical Committee Member and a Reviewer for the IEEE ICC, the IEEE CCNC, the IEEE LCN, the IEEE ACCESS, the Elsevier Computer Communications, and the IEEE Transactions on Network and Service Management, among others.



Thi-Minh Nguyen received her Ph.D. in Networks and Performance Analysis at Sorbonne Université (formally UPMC), France, in 2017. She was a Postdoctoral Researcher at LIP6, France, in 2018. She is currently working as a research engineer for Siemens France. She is also a lecturer at Dong Nai Technology University. Her research focuses on optimization problems in NFV infrastructure. She is particularly interested in mathematics and graph theory. She holds a MSc and BSc in Computer Science both from the Hanoi University of Science and Technology, Vietnam in 2013.