Dynamic Network Slicing Orchestration for Remote Adaptation and Configuration in Industrial IoT

Luyue Ji, *Student Member, IEEE,* Shibo He, *Senior Member, IEEE,* Wenjie Wu, *Student Member, IEEE,* Chaojie Gu, *Member, IEEE,* Jichao Bi, *Member, IEEE,* and Zhiguo Shi, *Senior Member, IEEE*

***Abstract*—As an emerging and prospective paradigm, the Industrial Internet of Things (IIoT) enables intelligent manufacturing through the interconnection and interaction of industrial production elements. The traditional approach that transmits data in a single physical network is undesirable because such a scheme cannot meet the network requirements of different industrial applications. To address this problem, in this paper, we propose a network slicing orchestration system for remote adaptation and configuration in smart factories. We exploit Software-Defined Networking (SDN) and Network Functions Virtualization (NFV) to slice the physical network into multiple virtual networks. Different applications can use a dedicated network that meets its requirements with limited network resources with this scheme. To optimize network resource allocation and adapt to the dynamic network environments, we propose two heuristic algorithms with the assistance of Artificial Intelligence (AI) and the theoretical analysis of the network slicing system. We conduct numerical simulations to learn the performance of the proposed algorithms. Our experimental results show the effectiveness and efficiency of our proposed algorithms when multiple network services are concurrently running in the IIoT. Finally, we use a case study to verify the feasibility of the proposed network slice orchestration system on a real smart manufacturing testbed.**

***Index Terms*—Network slicing, Industrial Internet of Things, Software-Defined Networking, artificial intelligence, remote adaptation and configuration.**

I. Introduction

Recently, Industry 4.0 is considered a significant driving force for the new generation of the industrial revolution, which is conducive to enter the era of intelligence. IIoT has aroused widespread attention, which satisfies various application scenarios such as remote adaptation and configuration, intelligent operation and maintenance, industrial Augmented Reality, digital twins, and equipment collaborative operation [1]–[4]. IIoT connects ubiquitous physical entities with computing capabilities based on Internet technology and standards to drive production with information and data technology through industrial data modeling, management, and analysis.

IIoT involves various vertical industries, such as smart manufacturing [5], autonomous driving [6], smart cities [7], and intelligent health [8]. With the continuous transformation of business and the emergence of new applications, their network demand fluctuates constantly in an orderly or disorderly manner. The conventional one-fits-all network construction method leads to expensive operation, deployment costs, and inefficient resource utilization. Therefore, how to accommodate the diversity and dynamics of business requirements in IIoT remains a challenging issue.

As a new generation of mobile communication technology, 5G effectively meets the high-performance and flexible networking requirements of the IIoT, with its extremely high transmission speed, strong connection capability, and real-time communication capability close to the industrial bus. Network slicing [9] is a key technology of 5G that refers to building a customized and isolated logical end-to-end network on the substrate network, providing differentiated network services for different user groups through various combinations of functions, performance, and network connection relationships. Meanwhile, it enables rapid network deployment and upgrades to accommodate to changing market demands. It is worth mentioning that network slicing can safely and effectively isolate different tenants, e.g., enterprise users who rent communication infrastructure from Internet Service Providers (ISPs), and they can fully participate in the life cycle management of slice instances.

The network slicing is implemented with the integration of two emerging technologies, SDN [10] and NFV [11], [12]. These two technologies empower end-to-end on-demand network slicing with flexibility and effectiveness. The SDN is a novel network architecture that adopts a centralized control scheme, in which the network can manage and optimize the network resources globally [13]. Traditionally, the network functions have to run on dedicated hardware, which is highly undesirable because the network functions can hardly scale up or migrate discretionarily. To decouple the hardware and software, NFV virtualizes and packages the network functions as virtual machines or containers. These network functions are called virtual network functions (VNFs), such as firewall, data processing, IP address management, etc. At the run time, each network service chains multiple VNFs on standard physical facilities.

While there have been many attempts in both academia and industry to exploit network slicing to improve the IIoT performance and resilience in smart manufacturing, including architecture design and implementation, resource optimization, and slicing orchestration, their performance gain may degrade since the nature of the network service are dynamic and diverse [14]. Moreover, it is important to explore how to satisfy

This work was supported by the National Key R&D Program of China under grant No. 2018YFB1702100, NSFC under grant No. U1909207, the Major Key Project of PCL under Grant PCL2021A09, and 5G Open laboratory of Hangzhou Future Sci-Tech City. *Corresponding author: Shibo He*

L. Ji, S. He, J. Wu, C. Gu, J. Bi, and Z. Shi are with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, Zhejiang, 310027, China. E-mail: {july_fish, s18he, wuwenj, gucj, jonny.bijichao, shizg}@zju.edu.cn.

a new network service as it emerges in a smart manufacturing system.

In this paper, we propose a network slicing architecture for remote adaptation and configuration in smart factories. Based on SDN and NFV, we further design a dynamic network slice orchestration system for smart manufacturing test bed, which supports multi-intention and multi-requirement network services including Remote Operation (RO), real-time Industrial Data Monitoring (IDM), and Video Surveillance (VS). To minimize the deployment cost and meet the deployment time requirement of network services, we propose a static network slicing algorithm to compute the optimal deployment strategy when the transmission link bandwidth and computing resources are limited. To further adapt to the dynamic changes of network services, we develop a rapid scaling strategy for network resources with the assistance of AI, i.e., an evolutionary algorithm. The evolutionary algorithm, which is a generic population-based meta-heuristic optimization algorithm based on Darwinian evolution, can capture global solutions of complex optimization problems [15], [16]. Specifically, we accommodate the evolutionary algorithm to smart manufacturing and integrate it into our orchestration system. To the best of our knowledge, this is the first network slicing system in IIoT that provides a global view of the IIoT system to utilize the network slicing fully.

The main contributions of this work are summarized as follows:

- To adapt various transmission requirements of network services in IIoT, we design a dynamic network slicing orchestration system based on SDN and NFV.
- Considering the dynamics and diversity, we formulate a minimum cost deployment problem subject to resource and delay constraints. Then, we propose two algorithms to optimize the network slicing strategy.
- Through numerical simulation, we demonstrate the proposed algorithms well solve the minimum deployment cost problem. We implement and verify the proposed system on a real test bed, which achieves on-demand network resource allocation and scaling.

The rest of this paper is organized as follows. Sec.II briefly reviews the related literature. Sec.III introduces the network slicing architecture for remote adaptation and configuration and build the mathematical model for the minimum deploy cost problem. Sec.IV proposes effective algorithms for resource allocation in a network slicing system. Sec.V evaluates the performance of the proposed algorithms. Sec.VI verifies the feasibility of our system on an intelligent manufacturing testbed. Sec.VII concludes the paper.

II. Related Work

Existing studies related to network slicing can be broadly divided into two categories, including the network slicing architecture design and implementation, and resources allocation and slicing orchestration. In the following subsection, we introduce the related works in detail.

A. Network slicing architecture and implementation

There are several architectures and frameworks presented and realized. For example, machine learning-based automatic network slicing framework was proposed in [17] to intelligently scale slice according to network state, trading off robustness and resource efficiency. Based on the Lightweight Slice Defined Cloud, Dantas et al. [18] proposed the Novel Enablers for Cloud Slicing project to provide intelligent orchestration for federated cloud infrastructures. To satisfy requirements of integrated fronthaul/backhaul transport network in a multi-tenant environment, 5G-Crosshaul architecture was designed [19] and implemented in [20], beneficial to the performance efficiency as well as supporting simultaneous use by different tenants. 5G NORMA [21] is a multi-service and multi-tenant 5G system architecture, which adapts to the requirements of heterogeneous services, mobility management, Quality of Service (QoS), control and orchestration. Capitani et al. [22] presented the experimental demonstration of a 5G network slice deployment through the 5G-transformer architecture, transforming rigid mobile transport networks into a flexible SDN/NFV-based mobile transport and computing platform supporting different verticals (e.g., automotive, e-health, e-industry). In order to shorten the time to market for new network services, Twamley et al. [23] proposed 5GTANGO utilized for the validation and verification of virtual network functions and network services. The numerous works presented multi-tenant network slicing architecture. However, more attention should be paid to the network slicing system in IIoT physical platforms, which considers various network services and heterogeneous industrial protocols/equipment in smart factories.

B. Resource allocation and slicing orchestration

Dawaliby et al. [24] proposed a distributed slicing scheme including slicing admission control, resource reservation and allocation, based on coalitional game and matching theory over an SDN-based LoRaWAN architecture. However, due to real-time network changes and different performance requirements between different 5G scenarios, network slices need to be adaptively adjusted online to adapt to the needs of specific services dynamically. Hence, Afolabi et al. [25] proposed a network slicing orchestration system, which includes a dedicated entity for each domain to manage the corresponding mobile network. Meanwhile, a dynamic auto-scaling algorithm was designed for adapting dynamic resource demands including proactive and reactive resource provisioning techniques based on G/G/m queue model of the network. The performance may be severely degraded when multiple VNFs are running on the same physical device. In order to solve it, a time-slot based 5G network slice model and an online lazy-migration adaptive interference-aware algorithm were proposed for real-time VNF deployment and cost-efficient VNF migration in a 5G network slice [26]. On the other hand, predictive scheduling is conducive to solving conflicts between service function chains of different network requests and improving server resource scheduling efficiency. Huang et al. [27] proposed POSCARS, an efficient predictive and online service chaining

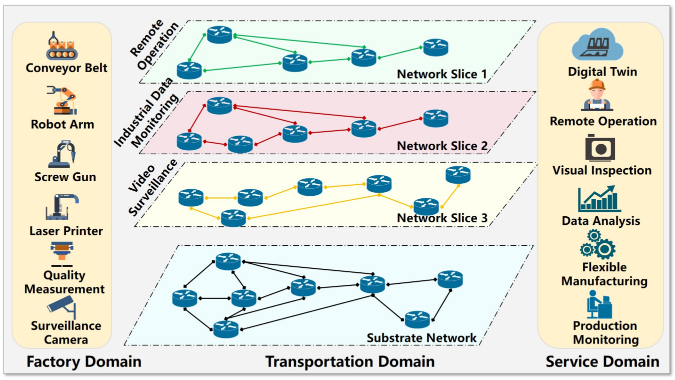


Fig. 1. The network slicing architecture for remote adaptation and configuration in a smart factory.

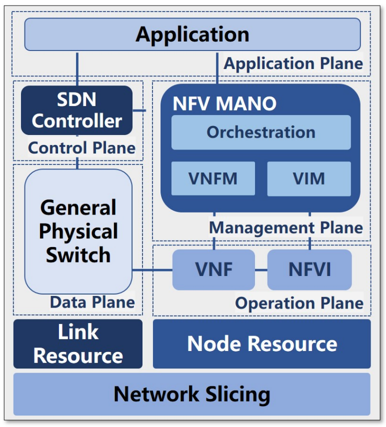


Fig. 2. The framework and components of network slicing system.

and resource scheduling scheme that achieves tunable trade-offs among various system metrics with stability guarantee. By a non-trivial transformation, the complex optimization problem was decoupled into a series of online sub-problems to achieve the optimality with only limited information.

Although the massive amount of network data puts tremendous pressure on network slicing performance, it also provides an opportunity for new resource management methods. AZTEC [28] is a data-driven framework, integrates the deep learning architectures and a traditional optimization algorithm to effectively allocates network capacity to individual slices. AZTEC predicts resource allocation to minimize the management costs caused by resource overprovisioning, instantiation and reconfiguration. A scalable digital twin of network slicing was developed [29], aiming to capture the intertwined relationships among slices and monitor the end-to-end metrics of slices under diverse network environments. The novel Graph Neural Network model was exploited to learn insights directly from slice-enabled networks represented by non-Euclidean graph structures. To solve the problem of reduced QoS caused by traffic surge and user mobility, Chen et al. [30] came up with a deep-learning-based multivariate long short term memory model to capture the spatiotemporal patterns of traffic and mobility for accurate prediction. The cost and quality objectives were formulated as a RRH-BBU mapping of the set partition problem, and Resource Constrained Label Propagation (RCLP) algorithm was proposed to solve it. The above studies focus on resource allocation of network slices. In this paper, we further consider the impacts of three factors, including network services requirements, communication resource, and computing resource.

III. System design and model

This section introduces the design and components of the dynamic network slicing orchestration and management system for remote adaptation and configuration of smart factories. We formulate the network slicing orchestration problem as a deployment cost optimization problem subject to the constraints of bandwidth resources, computing resources, and delay requirements.

A. System overview

In this paper, we define the remote adaptation and configuration of smart factories as a set of multi-intention services including RO, real-time IDM, and VS. Operators can accurately control the on-site industrial equipment remotely in real-time according to the video pictures and various data of the production site. Meanwhile, the intelligent detection system realizes safety monitoring and management of the production site through real-time industrial data collection, image recognition, custom alarms, and other technologies. Under this circumstance, different network services have varying requirements for bandwidth, delay, and network functions. To address this challenge, we exploit network slicing to virtualize multiple virtual networks (i.e., network slices) on one physical network to support multi-intention services simultaneously.

The network slicing architecture for the remote adaptation and configuration scenario, as shown in Fig.1, contains the factory domain, transportation domain, and service domain. The factory domain includes various types of sensors, actuators, smart equipment, and industrial instrumentation, with different data types, communication protocols, operating systems, and production beats. The service domain comprises of various intelligent applications, such as digital twin, remote operation, visual inspection, and production monitoring, with distinct communication constraints. The network slicing of the transportation domain enables scalable network resource configurations, satisfying different requirements of businesses and user groups.

B. Network slicing system architecture

It is desirable to have a resilient and tailored network that supports the concurrent transmission of multiple network services (e.g., remote control and monitoring). To this end, as shown in Fig. 2, we propose an SDN and NFV-based dynamic network slicing orchestration and management framework in the transmission domain. Based on a standard for NFV [31], we design the NFV architecture and integrate the SDN so that we can take advantage of both. As two core components, the SDN controller and NFV MANO cooperatively manage hardware resources to achieve centralized optimization and network parameter configuration. The system has six components, i.e., SDN Controller, General Physical Switch, NFV management and orchestration (NFV MANO), VNF, Network Functions Virtualization Infrastructure (NFVI), and Application. Next, we introduce each component from top to bottom.

1) Application: Applications exist in the application plane and interact with the SDN controller through northbound interfaces. Controlling the network through applications brings programmability, powerful network recovery capabilities, and rapid deployment mechanisms. It solves the limitations of traditional manual processing and greatly improves the flexibility and scalability of the network.

2) SDN Controller: The control plane is composed of one or more SDN controllers, which are logically centralized and have global knowledge of the network. The SDN controller is responsible for transmitting the decision information to the network equipment (e.g., general physical switches in the following) in southbound interfaces, such as OpenFlow, P4, OF-Config, NET-CONF, etc. In addition, the SDN controller first retrieves network status information, and then transmits processed information to NFV MANO, including the network traffic statistics, available bandwidth, and bandwidth usage efficiency, to optimize the life cycle management of VNFs.

3) General Physical Switch: General physical switches (i.e., programmable switches) constitute the data plane and forward user data according to the forwarding entries generated by the control plane. Switches in the data plane forward user data according to the forwarding entries generated by the control plane. NFVI and VNF modules are also based on the general physical switches. Virtual machines or containers are installed in their operating system to achieve specific VNFs thanks to the universal standard devices.

4) NFV MANO: NFV MANO contains Virtualized Infrastructure Manager (VIM), Virtualized Network Function Manager (VNFM), and NFV Orchestration. VIM manages hardware resources such as computing and storage, and supports the software and hardware of the virtualization layer in NFVI. VNFM creates VNF and manage the proportion of resources with the resource usage information from VIM. NFV Orchestration obtains the global topology knowledge and network status information from the SDN controller and conducts VIM and VNFM to deploy VNF instances and allocate infrastructure resources.

5) NFVI: NFVI is the foundation of hardware virtualization and treats the commercial off-the-shelf hardware as a public resource pool. NFVI divides resources into multiple subsets and creates virtualized computing, storage, and network resource pools according to VNF allocation requirements.

6) VNF: VNF module implements specific network functions by loading software on virtual machines achieved by NFVI. And network services require VNFs to process data packets in a particular order called a VNF-Forwarding-Graph (VNF-FG) or Service Function Chain (SFC).

C. System model

As mentioned in section III-A, there are various services in the remote adaptation and configuration scenario. The n^{th} network service $f_n \in \mathcal{F}$ is a five-tuple, i.e., $NS_n\left(s_n, d_n, SFC_n, r_n, D_n\right)$, which respectively represent the source node, destination node, service function chain, required bandwidth and end-to-end delay requirements. $\mathcal{F} = \{f_1, f_2, ..., f_n, ..., f_N\}$ represents the set of network services, and $N = |\mathcal{F}|$ represents the number of network services.

Clearly, different SFCs represent different network services. This is because the streams of different network services that go through the SFC consist of the specific order or quantities of VNFs according to their particularity. For instance, traffic of VS passes the firewall to filter traffic from limited IP addresses, video compression to mitigate the amount of data transferred, and video decompression to recover the original video. The real-time data of IDM go through the firewall, encryption and decryption for secure transmission.

The service function chain of network services f_n has J_n VNFs, $SFC_n = (VNF_1^n, ..., VNF_{J_n}^n)$. There are a total of M VNFs in the network. Each VNF_m has its specific hardware resources demand, $Demand(CPUCores, RAM, Disk)$, respectively indicating the required number of CPU cores, RAM and disk resources for the corresponding virtual machine. Note that the resources like RAM and Disk are abundant compared with CPU cores. Therefore, we consider only CPU cores as the resource constraint in physical switches.

We consider a physical network $G\left(V, L\right)$, where V represents the set of nodes, and L represents the set of directed links, $\left(u, v\right) \in L$. Our goal is to find a shortest path for the flow f_n to meet the transmission delay requirement of the network and service chain, and minimize the network cost with limited bandwidth and hardware resources. For a link between node u and v, the bandwidth constraint can be expressed as

$$\sum_{n=1}^{N} x_n\left(u, v\right) \cdot r_n \leq R\left(u, v\right), \forall\left(u, v\right) \in L, \tag{1}$$

where r_n denotes the bandwidth allocated to f_n and $R\left(u, v\right)$ is bandwidth capacity of the link. $x_n\left(u, v\right)$ is a binary variable. If the link between node u and v is allocated to f_n, $x_n\left(u, v\right) = 1$. Otherwise, $x_n\left(u, v\right) = 0$. Excepting s_n and d_n, the flow conservation constraint requires each physical node to satisfy that the number of inflows equals to the number of outflows, which is expressed as

$$\sum_{u=1}^{|V|} x_n\left(u, v\right) - \sum_{u=1}^{|V|} x_n\left(v, u\right) = \begin{cases} -1, & if\ v = s_n \\ 1, & if\ v = d_n \\ 0, & otherwise \end{cases}. \tag{2}$$

Similarly, since the number of CPU cores on a physical equipment is finite, the computing resource constraint is

$$\sum_{n=1}^{N} \sum_{j=1}^{J_n} y_n^{j,v} \cdot Q(VNF_{j,v}^n) \leq res_v, \forall v \in V, \tag{3}$$

where $y_n^{j,v}$ indicates the distribution of VNF, $Q(VNF_{j,v}^n)$ is the number of CPU cores required by the $VNF_{j,v}^n$, and res_v denotes the computing resource capacity of node v. If the j^{th} VNF of f_n is deployed at physical switch node v, $y_n^{j,v} = 1$. Otherwise, $y_n^{j,v} = 0$. When there is no link between node u and node v, the distribution of the VNF does not exist, i.e., $y^{j,v} = 0$, $y^{j,u} = 0$. And the VNF allocation constraint ensures that each VNF of network services is assigned to only one physical node, which means

$$\sum_{v=1}^{|V|} y_n^{j,v} = 1, \forall n \in N, j \in J_n. \quad (4)$$

In addition, IIoT requires real-time network services, so that the end-to-end delay requirements is expressed as

$$\sum_{(u,v)\in P_n} d_n^{u,v} + \sum_{v\in P_n} d_n^v \le D_n, \quad (5)$$

where

$$d_n^{u,v} = d_{u,v}^{trans} + d_{u,v}^{prop}, \quad \forall (u,v) \in P_n, \quad (6)$$

and

$$d_n^v = d_v^{queue}(VNF_j^n) + d_v^{proc}(VNF_j^n), \quad \forall v \in P_n. \quad (7)$$

Especially, the path of f_n is $P_n = \{v_{n,1}, v_{n,2}, ..., v_{n,l}\}$. $d_{u,v}^{trans}$ and $d_{u,v}^{prop}$ are the transmission delay and propagation delay on the link (u,v), respectively. $d_v^{queue}(VNF_j^n)$ and $d_v^{proc}(VNF_j^n)$ represent the queue delay and processing delay deploying VNF_j^n on node v, respectively, and the total delay can be regarded as the duration of f_n flowing through a certain VNF at a certain node.

The length of packet $L\ bits$ and allocated bandwidth r_n are fixed, hence $d_{u,v}^{trans} = \frac{L}{r_n}$ is a constant. To simplify the problem, we assume that r_n is large enough, so that $d_v^{queue}(VNF_j^n)$ can be ignored. The propagation delay and process delay are determined by the speed of light and the length of the cable and the resources allocated to the corresponding VM, respectively. Hence, both $d_{u,v}^{prop}$ and $d_v^{proc}(VNF_j^n)$ can be regarded as a constant. Therefore, $d_n^{u,v}$ and d_n^v can be obtained through experimental measurements. Intuitively, the time delay constraint inequality is actually a constraint on the number of path hops. Deployment cost is composed of node cost and link cost, where the node cost is the sum of the cost of computing resource deployment for all nodes and the link cost is the sum of the cost of bandwidth resource allocation for all links in P_n. The deployment cost to satisfy f_n is

$$\begin{aligned} cost_n &= \sum_{(u,v)\in P_n} cost_n^{u,v} + \sum_{v\in P_n} cost_n^v \\ &= \sum_{(u,v)\in P_n} c_{u,v} r_n + \sum_{v\in P_n} c_v(VNF_j^n), \quad \forall n \in N, \end{aligned} \quad (8)$$

where $cost_n^v$ represents the node cost required to satisfy f_n, $cost_n^{u,v}$ represents the link cost, $c_{u,v}$ is the cost required to allocate unit bandwidth, and $c_v(\cdot)$ represents the cost function of resources required to deploy VNF_j^n on node v, which is a monotone increasing function. In general, when a new network service appears in the network, we need to determine whether the current network has sufficient resources. If available resources are adequate, the network service is approved to access the network. The optimal path of the minimum deployment cost is computed for the network service while meeting bandwidth and computing resources constraints and end-to-end delay requirements.

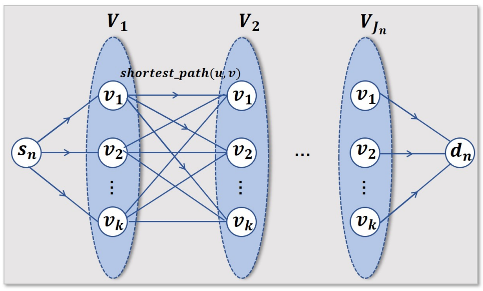


Fig. 3. Construction of the auxiliary graph H_n for the network service f_n.

IV. Optimization of the Network Slicing System

This section proposes a static network resource allocation (STNRA) algorithm to obtain the optimal slicing strategy with relatively stable network conditions. In addition, considering that the demand for network services changes dynamically in the real smart factory, we propose a dynamic network slicing orchestration (DNSO) algorithm for such a scenario.

A. Static network resource allocation

Our proposed problem is regarded as two integrated selection schemes, including path selection for the traffic transmission and node selection for the VNFs deployment. Generally speaking, the delay-constrained shortest path problem is NP-hard. And then, the path selection problem with VNF selection is still an NP-hard problem [32]. Therefore, in this subsection, we transform the original problem into a delay-constrained shortest path problem by constructing an auxiliary graph [32] for each network service, and then mainly focus on solving the proposed optimization problem by applying Lagrange Relaxation based Aggregated Cost (LARAC) algorithm [33] to compute the least costly path that meets the delay requirement and limited resources.

1) Construction of the auxiliary graph: H_n denotes the auxiliary graph for each incoming network service f_n. The auxiliary graph is composed of the source node, destination node, and all physical switch nodes of the network service. As shown in Fig.3, there are three steps to build an auxiliary graph. Firstly, we deploy J_n VNF sets in order according to the SFC information of the f_n, and each set V_j^n consists of all physical nodes, i.e., $V_j^n = V$. Secondly, we denote the edge weights of the source node to VNF_1^n and the destination node to VNF_c^n as $shortest_{path}(s_n, VNF_1^n)$ and $shortest_{path}(VNF_{J_n}^n, d_n)$, respectively. Thirdly, we compute the shortest paths between nodes $u \in V_j^n$ and $v \in V_{j+1}^n$, and denote the edge weights as the cost of the shortest path between them in $G(V,L)$ if the shortest path exists[1].

2) STNRA algorithm for minimizing the deployment cost of slices: As shown in Algorithm 1, we first compute shortest paths of all pairs, (i.e., $shortest\_path(u,v), \forall u, v \in V$) in $G(V,L)$ and construct the auxiliary directed acyclic graph H_n. Based on LARAC Algorithm, we obtain the shortest path P_n, the indicator variables $x_n(u,v)$ and $y_n^{j,v}$ with the delay

[1]The weight is recorded as positive infinite if the shortest path does not exist.

Algorithm 1 STNRA algorithm for minimizing the deployment cost of slices

Input: Set of network services $\mathcal{F}$, the physical network $G(V, L)$.

Output: The shortest path P_n^*, the optimal indicator variables $x_n^*(u,v)$ and $y_n^{*j,v}$.

```
1:  Initialization: the number of network services $N := |\mathcal{F}|$, bandwidth capacity $R(u,v)$, delay of link $d_n^{u,v}$, delay of node $d_n^v$;
2:  $n = 0$, $FLAG = 1$;
3:  while $n \leq N$ do
4:      Compute all pairs shortest paths in $G(V, L)$, construct the auxiliary directed acyclic graph $H_n$ and assign a weight to each of its nodes and links by Eq. (8);
5:      Find a shortest path $P_n$ in $H_n$ for $f_n$ with the delay requirement by invoking LARAC Algorithm, and obtain the $x_n(u,v)$ and $y_n^{j,v}$;
6:      if $\sum_{n=1}^{N} x_n(u,v) \cdot r_n \leq R(u,v), \forall (u,v) \in L$ and $\sum_{n=1}^{N} \sum_j y_n^{j,v} \leq res_v, \forall v \in V$ then
7:          $P_n^* = P_n$, $x_n^*(u,v) = x_n(u,v))$, $y_n^{*j,v} = y_n^{j,v}$;
8:      else
9:          Delete the corresponding link or node in $G(V, L)$;
10:     end if
11:     $n = n + 1$;
12: end while
```

Algorithm 2 DNSO algorithm to optimize the deployment cost of slices

Input: Set of network services $\mathcal{F}$, the physical network $G(V, L)$.

Output: The $|V|*|V|$ path selection matrix X_n, the $J_n*|V|$ VNF allocation matrix Y_n, the optimal auxiliary vector $\mathbf{z}^*$, the minimum deployment cost $cost_n{}^*$.

```
1:  Initialization: the number of network services $N := |\mathcal{F}|$, bandwidth capacity $R(u,v)$, delay of link $d_n^{u,v}$, delay of node $d_n^v$;
2:  while $n \leq N$ do
3:      Dimensionality reduction through the variable mapping $(X_n, Y_n) \rightarrow \mathbf{z}$;
4:      Compute the optimal $\mathbf{z}^*$ and $cost_n{}^*$ based on NAA subject to the constraint (1) - (5);
5:      while $order_n$ is FALSE do
6:          Add $\mathbf{z}((X_n, Y_n))! = \mathbf{z}^*(X_n{}^*, Y_n{}^*)$ as a new constraint to the original problem;
7:          Recompute the optimal $\mathbf{z}^*$ based on NAA;
8:      end while
9:      $n = n + 1$;
10: end while
```

requirement (line 4-5). Since the bandwidth of the physical link and the computing resources of the physical switch are limited, if the resource constrains (Eq. 1 and Eq. 3) are violated, i.e., $\sum_{n=1}^{N} x_n(u,v) \cdot r_n \geq R(u,v), \forall (u,v) \in L$ or $\sum_{n=1}^{N} \sum_j y_n^{j,v} \geq res_v, \forall v \in V$, we will delete the corresponding link or node in $G(V, L)$ and re-compute the auxiliary graph. Otherwise, we obtain the resource allocation strategy $(P_n^*, x_n^*, y_n^{*j,v})$ of the network service f_n (line 6-10).

B. Dynamic network slicing orchestration

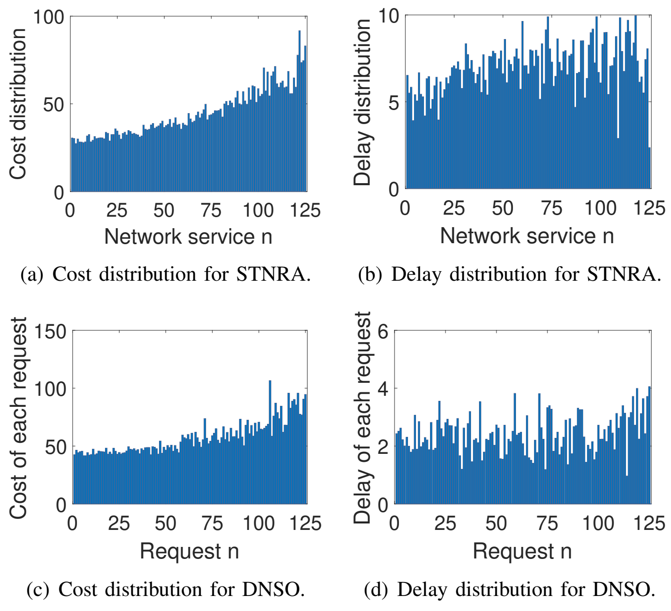
In a smart factory, the mobility of equipment and changes in perception strategies and production tasks trigger network traffic to change in both space and time domains continuously. The STNRA algorithm based on graph theory recursively calculates the auxiliary graph for network services. As the computation cost increases with the computation space, we leverage the Natural Aggregation Algorithm (NAA) [34] to implement the DNSO algorithm. NAA is an emerging evolutionary algorithm inspired by the collective decision-making intelligence of group-living animals. We further accommodate the NAA to multi-service industrial scenarios.

1) Dimensionality reduction: The $(|V| + J_n) * |V|$ variables have an extremely high computational complexity. And the path selection matrix X_n and VNF allocation matrix Y_n are sparse matrices, making most of the computing power wasted. To improve the computation efficiency, we construct an auxiliary vector $\mathbf{z} = [p_1, p_2, ..., p_M, q_1, q_2, ..., q_{J_n}, s]$ in DNSO algorithm. Especially, $M \leq |V|$ denotes the expected length of the selected path, q_j indicates the deployment location of the VNF, p_i and s denote the node and valid length of the selected path, respectively. The first s nodes of the path are valid, and the remaining nodes are set to zero. For example, for a network service with 2 VNFs in the network of 6 nodes, $\mathbf{z} = (1, 3, 5, 0, 0, 0, 1, 3, 3)$ denotes that the transmission path of the data flow is $v_1 \rightarrow v_3 \rightarrow v_5$, the two VNFs are deployed on node v_1 and v_3 in order, and the first 3 nodes of the path are valid. Based on this method, we map the original variables to $\mathbf{z}$ decreasing the number of variables from $(|V| + J_n) * |V|$ to $(M + J_n + 1)$.

2) DNSO algorithm: Algorithm 3 describes how to optimize the deployment cost of network slicing. We first conduct dimensionality reduction exploiting the auxiliary vector $\mathbf{z}$. The NAA is invoked to search for the optimal path selection and VNF deployment strategy subject to the constraint (1)-(5) (line 4). NAA generates a population of D-dimension vector, which representing potential solutions for the given problem. The algorithm evolves according to the constraints and optimization objectives set, and improves the accuracy by adjusting the number of iterations and clusters. Then we judge whether the scheme satisfies the SFC sequence constraint, that is, whether the VNF is deployed in a specific order. If satisfied, the current plan is the optimal strategy. Otherwise, this strategy should be eliminated and recomputed until the constraint is met (line 5-8). The time complexity of algorithm 3 is $O(N \times P \times I \times D)$, where P is the number of populations, I is the number of iterations, and $D \leq (M + J_n + 1)$ is the dimension of optimization variable.

V. Performance evaluation

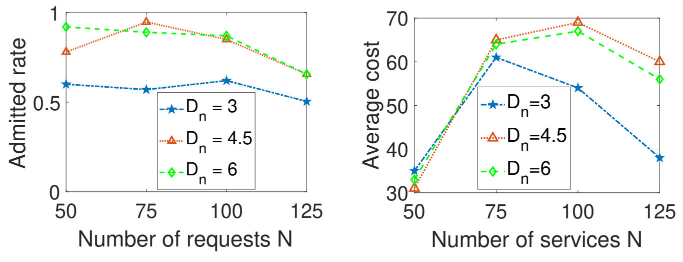
In this section, we profile the performance of STNRA and DNSO algorithm through MATLAB numerical simulation,



(a) Cost distribution for STNRA. (b) Delay distribution for STNRA.

(c) Cost distribution for DNSO. (d) Delay distribution for DNSO.

Fig. 4. Status distribution of each network service, when $N = 125$.



(a) Admitted rate. (b) Average deployment cost

Fig. 5. Effects of delay requirement on network services for STNRA, when $V = 50$.

including the efficiency of the network slicing orchestration scheme, end-to-end network delay, and service acceptance rate. Furthermore, we compare and evaluate the performance of our solution for different parameters, such as network services number N, the cost function of resources required to deploy VNFs, bandwidth capacity $R(u, v)$, and end-to-end delay requirement D_n, respectively. We also compared the performance of STNRA and DNSO algorithm.

We conduct simulations on a laptop, which has a 1.6 GHz CPU and 8 GB RAM. Unless otherwise stated, the parameters used in the simulation are set as follows. In our experiments, network size is $|V| \in \{50, 75, 100, 125\}$ and the number of network services is $N \in \{50, 75, 100, 125, 150\}$ [26]. The bandwidth capacity of the link $R(u, v)$ is 100 Mbps. Considering that VS has high requirements for bandwidth, while RO and IDM are sensitive to delay. For RO, the delay requirement D_n is $0.1 \sim 0.25$ ms, and the required bandwidth r_n is $0.5 \sim 1$ Mbps. For IDM, D_n is $3 \sim 5$ ms and r_n is $1 \sim 5$ Mbps. For VS, D_n is $10 \sim 15$ ms and r_n is $20 \sim 40$ Mbps. The number of CPU cores of a single physical node is 8. The number of VNF in SFC_n is $J_n \in \{3, 4, 5, 6, 7\}$ and the required number of CPU cores is 1 or 2. According to different application scenarios, the cost function could be an inverse proportional function, a step function or an exponential function.

A. Status distribution of each network service

In this subsection, we focus on the status distribution of each network service, including deployment cost and the end-to-end delay in the STNRA and DNSO algorithm. To compare the performance of the two algorithms, we normalize cost and delay for all results. Fig. 4(a) and Fig. 4(c) show that the deployment cost of network service increases as the serial number of the network services increases. This is because network services enter the network sequentially in our algorithm, which decreases the available resources for subsequent network services. With insufficient resources, network services may choose a longer path, increasing the deployment cost. Compared with DNSO, STNRA searches the values slowly but exhaustively, which results in a lower deployment cost. In other words, DNSO sacrifices the accuracy for its real-time performance. Fig. 4(b) and Fig. 4(d) depict that the end-to-end delay of the network service depends on the delay requirement of the corresponding network service, so the distribution of the end-to-end delay is relatively uniform. In addition, the average delay of DNSO is less than that of STNRA, attributing to the more allocated resources.

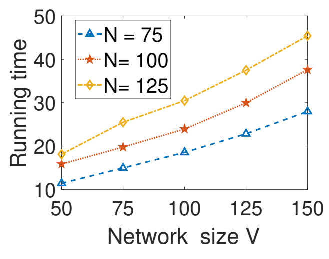
B. Impacts of delay requirement and network scale

This subsection measures how the delay requirement and network scale affect the performance in STNRA and DNSO algorithms on the network service. Different lines represent different delay requirements. In this experiment, we consider three delay requirements, i.e., $D_n = 3, 4.5, 6$ units. Fig. 5(a) shows the admitted rate of network services with different delay requirements for STNRA. The admitted rates decrease with the increasing the number of network services or enhancing end-to-end delay requirements. Fig. 5(b) shows that the average cost of the network increases as the number of network services increases when N is small. After the average cost reaches the peak value ($N = 100$), it decreases as the number of network services increases. We can also see that, a stricter delay requirement causes a lower admitted rate and a smaller average cost of the network, because the available bandwidth or CPU resources are insufficient in the network when N is too much, or D_n is excessively strict.

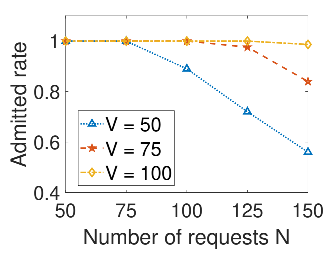
Fig. 6(a) and Fig. 6(b) describe the effects of network scale on network services for DNSO. We observe that the running time increases as the network scale (N and V) increases. Moreover, excessive network services or inadequate network sizes significantly decrease the admitted rates. Similar to before, superabundant network services lead to insufficient network resources.

C. Performance comparison

In this subsection, we firstly compare the deployment cost of our two algorithms with different cost functions. In

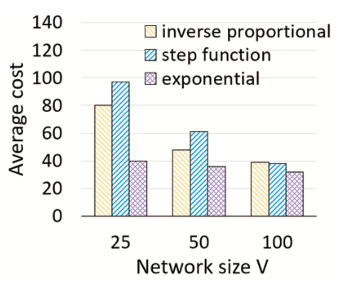


(a) Average Running time.

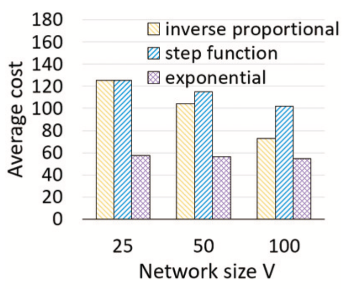


(b) Admitted rate.

Fig. 6. Effects of network scale on network services for DNSO.



(a) Average cost for STNRA.

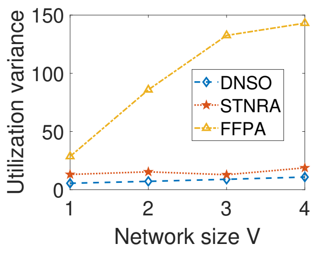


(b) Average cost for DNSO.

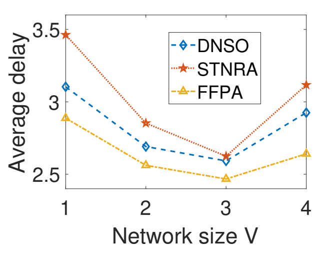
Fig. 7. Effects of cost function on network services

the simulation, the step-shape cost function is defined as $f(x) = \alpha_1 \lfloor \alpha_2 x \rfloor$, the inverse proportional cost function is $f(x) = \beta \cdot \frac{1}{res_v - x + 1}$, and the exponential cost function is $f(x) = \gamma_1 \cdot exp(\gamma_2(1 - \frac{x}{res_v}))$, where $\alpha_1 = 12$, $\alpha_2 = 1/5$, $res_v = 30$, $\beta = 5$, $\gamma_1 = 5$, and $\gamma_2 = 2$ are regulatory factors of cost functions determined by the physical network status and specific environment. In particular, we add 1 to $(res_v - x)$ so that the denominator of inverse proportional cost function (i.e., $res_v - x + 1$) is always greater than 0. From Fig. 7(a) and Fig. 7(b), we can see that the average cost of the network service decreases as network nodes increase regardless of the cost function. It is straightforward that the abundance of network resources reduces the deployment cost. Also, the step-shaped cost function of VNF deployment leads to higher network costs, and the exponential cost function reduces the average network cost. Besides, the DNSO algorithm generates more average cost due to the incomplete search.

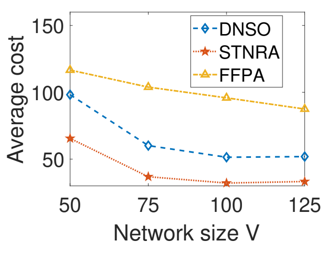
Fig. 8 presents the comparison between our two algorithms and First-Fit Placement Algorithm (FFPA) [35] in terms of computing resource utilization variance, average end-to-end delay, average deployment cost, and admitted rate. Since FFPA places each VNF at the first available node with adequate capacity, which considerably decreases the computation complexity of search algorithms, we consider FFPA as the baseline for performance comparison. With the identical network scale and transmission requirements, the strategy computed by STNRA causes the lowest average deployment cost but most end-to-end transmission time. In the DNSO scheme, network nodes' computing resource utilization variance is the smallest, which means the most balanced VNF deployment. Furthermore, DNSO has a smaller average cost and a higher admitted rate than FFPA. In conclusion, the performance of the



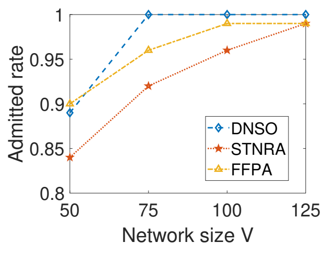
(a) Running time.



(b) Average E2E delay.



(c) Average cost.



(d) Admitted rate.

Fig. 8. Comparison of algorithms, when $N = 100$.

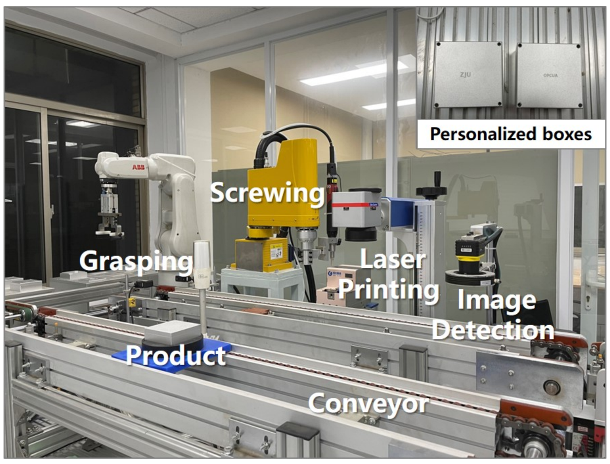


Fig. 9. The intelligent manufacturing testbed.

DNSO algorithm is superior in constantly changing industrial environments.

VI. Testbed and Case Study

In this section, we present an OPC Unified Architecture-based (OPC UA [36]) intelligent manufacturing testbed for remote adaptation and configuration. To meet the requirements of multiple network services, we implement the network slicing orchestration and management system on the testbed.

A. Production line

Fig. 9 shows that our experimental platform [36] takes the personalized box as the production target and extracts the common processing process of the discrete manufacturing industry, including the grasping unit, screwing unit, laser printing unit, and image detection unit. First of all, Robot 1 (ABB IRB 120 Robot [37]) grabs the base and cover of the box to the tooling plate, passes it through the conveyor belt to the screw-tightening unit, and then uses the Robot 2 (TURIN STH030-500 Robot [38]) with a screw shaft (Desoutter CVIC II ECSF10 [39]) to tighten the four screws to the box cover. After the assembly is completed, the laser printer (Kinglee F2000 [40]) prints character strings or patterns on the box. Then, the quality inspection of the assembly and marking results of the personalized box is carried out by a camera (Cognex In-Sight 7010C [41]) in the image detection unit. Finally, the finished box is transported to the pickup area through a circular conveyor belt. In addition, the testbed is equipped with four displacement sensors (Keyence DL-EPL IL-056 [42]) to detect the position of the product. The logic control of the entire production line is completed by the Programmable Logic Controller (PLC, MELSEC iQ-F FX5U-64MT [43]) through the ladder diagram.

B. *OPC UA-based communication and data collection*

There are various communication protocols in the testbed, including PROFINET [44], Modbus [45], EtherCAT [46], Ethernet/IP [47]. Note that some protocols are proprietary and not compatible with each other. We propose to use the OPC UA protocol to solve the incompatibility issue. The OPC UA server collects the equipment status data and stores them in a unified format. For example, the four-arm TURIN Robot uses the Modbus communication protocol. The OPC UA server parses Modbus messages and converts them in the form of the information model [36]. The information model is composed of nodes (i.e., object nodes, variable nodes, and method nodes) and their reference relationships. A variable or method node belongs to an object node. Specifically, a variable node represents a physical parameter of an object, and a method node delegates an operation controlling devices' status called by OPC UA clients. For instance, *FourArmRobot* and *LaserPrinter* are two object nodes on the OPC UA server, which represent the information models of TURIN Robot and Kinglee laser printer, respectively. The object node, *FourArmRobot*, has several variable nodes, such as *FourArmRobotX* and *FourArmRobotY*, indicating the current position coordinates of the X-axis and Y-axis, respectively. The OPC UA client can subscribe to variable nodes to obtain real-time or historical data of the production line. The *LaserPrinter* has a method node called *LaserPrinterPattern*, which allows users to customize the printed pattern. The OPC UA client can operate the equipment by calling method nodes.

C. *Case study*

In this subsection, we build the communication network for our intelligent manufacturing testbed to realize remote adaptation and configuration in IIoT. We virtualize three mutually isolated network slices (i.e., RO, real-time IDM, and VS), and there may be multiple network requests on each slice. RO and IDM are respectively implemented through the method node and variable node of the OPC UA information model of the production line equipment. Users can modify the speed of the conveyor belt, change the laser-printed patterns and monitor industrial data during the production process thousands of miles away. As for VS, we leverage a high-definition camera (EZVIV C6WI [48]) to monitor the running status in order to make timely adjustments and ensure a safe production environment.

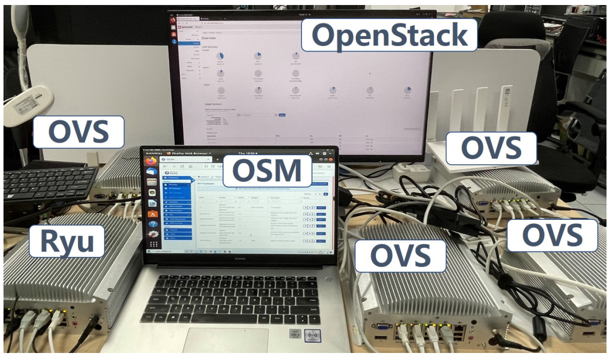


Fig. 10. The network slicing orchestration system in the physical network.

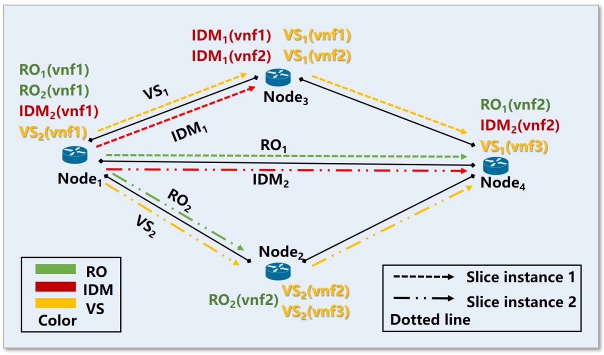


Fig. 11. The deployment scheme computed by the proposed algorithms.

In our experimental environment, we consider 2 RO, 2 IDM, and 2 VS requests and exploit 5 common standard industrial microcomputers (Shenzhen Konghui MFC-3102 [49]) as physical network nodes. As shown in Fig. 10, we deploy one Ryu controller (SDN controller), four Open vSwitches (OVSs, SDN switches), one OpenStack (VIM), and one Open Source MANO (OSM, NFV MANO) on microcomputers to implement the network slicing orchestration system.

Before we implement the physical network slicing, the minimum cost deployment scheme is obtained through numerical computations. The network scale in the experimental environment is relatively small, so we get the same deployment scheme (Fig. 11) computed by the STNRA and DNSO algorithm. Ryu controller allocates corresponding ports and appropriate bandwidth to six network slice instances. The VNFs are orderly deployed on the VMs created by OpenStack for the slice instances. In our physical network, the average end-to-end latency of the three slices all satisfy the demands of network services, which are 0.85 ms, 7.48 ms, and 461.33 ms, respectively.

VII. Conclusion

This paper presented the study on network slicing for the remote adaptation and configuration scenario in smart factories. Based on SDN and NFV, we implemented a dynamic network slice orchestration system, which supports multi-intention and multi-requirement network services, including RO, IDM, and VS. To minimize the deployment cost for network services, we further proposed two network slicing algorithms. Simulation results demonstrate the effectiveness and efficiency of our proposed algorithms when multiple network services are concurrently running in the IIoT. Implementation of the proposed network slice orchestration system on a real smart manufacturing testbed verifies the feasibility of our solution.

REFERENCES

[1] W. Mao, Z. Zhao, Z. Chang, G. Min, and W. Gao, "Energy efficient industrial internet of things: Overview and open issues," *IEEE Transactions on Industrial Informatics*, 2021.

[2] Y. Liu, X. Ma, L. Shu, G. P. Hancke, and A. M. Abu-Mahfouz, "From industry 4.0 to agriculture 4.0: Current status, enabling technologies, and research challenges," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4322–4334, Jun 2021.

[3] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation ethernet, iiot, and 5g," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 944–961, Jun 2019.

[4] K. R. Choo, S. Gritzalis, and J. H. Park, "Cryptographic solutions for industrial internet-of-things: Research challenges and opportunities," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3567–3569, Aug 2018.

[5] K. Tange, M. De Donno, X. Fafoutis, and N. Dragoni, "A systematic survey of industrial internet of things security: Requirements and fog computing opportunities," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2489–2520, 2020.

[6] M. Serror, S. Hack, M. Henze, M. Schuba, and K. Wehrle, "Challenges and opportunities in securing the industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 2985–2996, May 2021.

[7] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial internet of things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78 238–78 259, Dec 2018.

[8] W. Khan, M. Rehman, H. Zangoti, M. Afzal, and K. S. N. Armi, "Industrial internet of things: Recent advances, enabling technologies and open challenges," *Computers & Electrical Engineering*, vol. 81, p. 106522, Jan 2020.

[9] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5g network slicing using sdn and nfv: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167, p. 106984, Feb 2020.

[10] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging sdn and nfv security mechanisms for iot systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 812–837, 2019.

[11] A. J. Gonzalez, G. Nencioni, A. Kamisiski, B. E. Helvik, and P. E. Heegaard, "Dependability of the nfv orchestrator: State of the art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3307–3329, 2018.

[12] V. Nguyen, A. Brunstrom, K. Grinnemo, and J. Taheri, "Sdn/nfv-based mobile packet core network architectures: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.

[13] "Software-defined networking: The new norm for networks," White Paper, Open Networking Foundation, Apr 2012.

[14] S. Messaoud, A. Bradai, O. B. Ahmed, P. T. A. Quang, M. Atri, and M. S. Hossain, "Deep federated q-learning-based network slicing for industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5572–5582, Aug 2021.

[15] F. Luo, J. Zhao, and Z. Dong, "A new metaheuristic algorithm for real-parameter optimization: Natural aggregation algorithm," in *Proc. IEEE Congress on Evolutionary Computation (CEC'16)*. Vancouver, Canada: IEEE, Jul 2016, pp. 94–103.

[16] K. Matsuno, A. Ecer, N. Satofuka, J. Periaux, and P. Fox, "Parallel evolutionary computation for solving complex cfd optimization problems : A review and some nozzle applications," *Parallel Computational Fluid Dynamics 2002*, vol. 2003, pp. 573–604, 2002.

[17] F. Zhou *et al.*, "Automatic network slicing for iot in smart city," *IEEE Wireless Communications*, vol. 27, no. 6, pp. 108–115, Dec 2020.

[18] F. S. D. Silva *et al.*, "Necos project: Towards lightweight slicing of cloud federated infrastructures," in *Proc. IEEE Conference on Network Softwarization and Workshops (NetSoft'18)*. Montreal, QC, Canada: IEEE, Jun 2018, pp. 406–414.

[19] X. Costa-Perez *et al.*, "5g-crosshaul: An sdn/nfv integrated fronthaul/backhaul transport network architecture," *IEEE Wireless Communications*, vol. 24, no. 1, pp. 38–45, Feb 2017.

[20] X. Li *et al.*, "5g-crosshaul network slicing: Enabling multi-tenancy in mobile transport networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 128–137, Aug 2017.

[21] M. Gramaglia *et al.*, "Flexible connectivity and qoe/qos management for 5g networks: The 5g norma view," in *Proc. IEEE International Conference on Communications Workshops (ICC'16)*. Kuala Lumpur, Malaysia: IEEE, May 2016, pp. 373–379.

[22] M. Capitani *et al.*, "Experimental demonstration of a 5g network slice deployment through the 5g-transformer architecture," in *European Conference on Optical Communication (ECOC'18)*, Rome, Italy, Sep 2018, pp. 1–3.

[23] P. Twamley, M. Müller, P. B. Bök, G. K. Xilouris, C. Sakkas, M. A. Kourtis, M. Peuster, S. Schneider, P. Stavrianos, and D. Kyriazis, "5gtango: An approach for testing nfv deployments," in *2018 European Conference on Networks and Communications (EuCNC)*, Ljubljana, Slovenia, Jun 2018, pp. 1–218.

[24] A. B. S. Dawaliby and Y. Pousset, "Distributed network slicing in large scale iot based on coalitional multi-game theory," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1567–1580, Dec 2019.

[25] I. Afolabi, J. Prados-Garzon, M. Bagaa, T. Taleb, and P. Ameigeiras, "Dynamic resource provisioning of a scalable e2e network slicing orchestration system," *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2594–2608, 2019.

[26] Q. Zhang, F. Liu, and C. Zeng, "Online adaptive interference-aware vnf deployment and migration for 5g network slice," *IEEE/ACM Transactions on Networking*, 2021.

[27] X. Huang, S. Bian, X. Gao, W. Wu, Z. Shao, Y. Yang, and J. C. Lui, "Online vnf chaining and predictive scheduling: Optimality and trade-offs," *IEEE/ACM Transactions on Networking*, 2021.

[28] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "Aztec: Anticipatory capacity allocation for zero-touch network slicing," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 794–803.

[29] H. Wang, Y. Wu, G. Min, and W. Miao, "A graph neural network-based digital twin for network slicing management," *IEEE Transactions on Industrial Informatics*, 2020.

[30] L. Chen, T. M. T. Nguyen, M. N. D. Yang, C. Wang, and D. Zhang, "Data-driven c-ran optimization exploiting traffic and mobility dynamics of mobile users," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1773–1788, May 2021.

[31] "Network functions virtualisation," White Paper, European Telecommunications Standards Institute, Jan 2015.

[32] M. Jia, W. Liang, M. Huang, Z. Xu, and Y. Ma, "Routing cost minimization and throughput maximization of nfv-enabled unicasting in software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 732–745, Jun 2018.

[33] A. Juttner, B. Szviatovski, I. Mecs, and Z. Rajko, "Lagrange relaxation based method for the qos routing problem," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications*. Anchorage, AK, USA: IEEE, Apr 2001, pp. 859–868 vol.2.

[34] F. Luo, Z. Dong, Y. Chen, and J. Zhao, "Natural aggregation algorithm: a new efficient metaheuristic tool for power system optimizations," in *2016 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. Sydney, Australia: IEEE, Nov 2016, pp. 186–192.

[35] N. Kiran, X. Liu, S. Wang, and C. Yin, "Vnf placement and resource allocation in sdn/nfv-enabled mec networks," in *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2020, pp. 1–6.

[36] R. Wang, L. Ji, T. Ren, S. He, and Z. Shi, "A low-latency and interoperable industrial internet of things architecture for manufacturing systems," in *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, vol. 1, 2020, pp. 859–864.

[37] "Abb irb 120 robot," Asea Brown Boveri. [Online]. Available: https://new.abb.com/products/robotics/zh/industrial-robots/irb-120

[38] "Sth030-500 robot," TURIN Robot. [Online]. Available: https://www.turinrobot.com/scarxl/sth030500mm.shtml

[39] "Cvic ii ecsf10," Desoutter Industrial Tools. [Online]. Available: https://www.desouttertools.com.cn/gongju/2/systemes-electriques-dassemblage/25/cvic-ii-range/251/ecs-luo-si-dao/p/6151655050/ecsf10

[40] "F2000 laser printer," Kinglee. [Online]. Available: http://www.wxpenmaji.com/d320.html

[41] "In-sight 7010c," Cognex. [Online]. Available: https://www.cognex.com/blogs/machine-vision/easy-in-action-in-sight-7010

[42] "Dl-epl il-056 laser sensor," Keyence. [Online]. Available: https://www.keyence.com/products/measure/laser-1d/il/

[43] "Melsec iq-f fx5u-64mt," Mitsubishi Electric. [Online]. Available: https://no3a.mitsubishielectric.com/fa/no/dl/11209/JY997D55301-F.pdf

[44] R. Pigan and M. Metter, *Automating with PROFINET: Industrial communication based on Industrial Ethernet*. John Wiley & Sons, 2008.

[45] A. Swales *et al.*, "Open modbus/tcp specification," *Schneider Electric*, vol. 29, pp. 3–19, 1999.

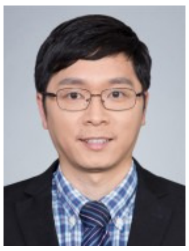
[46] D. Jansen and H. Buttner, "Real-time ethernet: the ethercat solution," *Computing and Control Engineering*, vol. 15, no. 1, pp. 16–21, 2004.

[47] Y. Tang, J. Qiu, Z. Tang, and X. Ling, *Fieldbus and Industrial Control Network*. China Machine Press, 2018.

[48] "Ezviv c6wi camera," Hikvision. [Online]. Available: https://www.ys7.com/item/165990.html?from=&position=cn

[49] Konghui, "Mfc-3102." [Online]. Available: http://www.gkj-eip.com/prod_view.aspx?nid=3&typeid=167&id=251

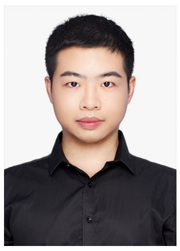
Luyue Ji (S'19) received the B.Eng. and M.Eng degrees from Southwest University, Chongqing, China, in 2012 and 2016, respectively. She is currently working toward the Ph.D. degree in control science and engineering with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China. Her research interests include industrial IoT, software-defined networking, network slicing, and resource allocation.



Shibo He (M'13-SM'19) received the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2012. He is currently a Professor with Zhejiang University. He was an Associate Research Scientist from March 2014 to May 2014, and a Postdoctoral Scholar from May 2012 to February 2014, with Arizona State University, Tempe, AZ, USA. From November 2010 to November 2011, he was a Visiting Scholar with the University of Waterloo, Waterloo, ON, Canada.

His research interests include Internet of Things, crowdsensing, big data analysis, etc. Prof. He serves on the editorial board for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, Springer Peer-to-Peer Networking and Application and KSII Transactions on Internet and Information Systems, and is a Guest Editor for Elsevier Computer Communications and Hindawi International Journal of Distributed Sensor Networks. He was a Symposium Co-Chair for the IEEE GlobeCom 2020 and the IEEE ICC 2017, TPC Co-Chair for i-Span 2018, a Finance and Registration chair for ACM MobiHoc 2015, a TPC Co-Chair for the IEEE ScalCom 2014, a TPC Vice Co-Chair for ANT 2013´lC2014, a Track Co-Chair for the Pervasive Algorithms, Protocols, and Networks of EUSPN 2013, a Web Co-Chair for the IEEE MASS 2013, and a Publicity Co-Chair of IEEE WiSARN 2010, and FCN 2014.

Wenjie Wu (S'21) received the B.Eng. degree from the School of Automation, Nanjing University of Aeronautics and Astronautics, China, in 2019, majored in detection guidance and control technology and studied the fault-tolerant control of small unmanned aerial vehicle. He is currently pursuing the master's degree in School of Engineering, Zhejiang University, China, with research interests about Network Function Virtualization, Software Defined Network, and cloud native technology.



Chaojie Gu (M'18) received the B.Eng. degree from Harbin Institute of Technology, Weihai, China, in 2016, and the Ph.D. degree in computer science and engineering from Nanyang Technological University, Singapore, in 2020. He was a Research Fellow with Singtel Cognitive and Artificial Intelligence Lab for Enterprise (SCALE) in 2021. He is an Assistant Professor with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China. His research interests include IoT, industrial IoT, edge computing, and low power wide area network.

Jichao Bi (S'19-M'20) is a Postdoctoral Research fellow with State Key Laboratory of Industrial Control Technology, Zhejiang University, China. He received Ph.D. degree in software engineering from Chongqing University, China, in 2020. He was a visiting Ph.D. student with the school of electrical engineering and telecommunications, University of New South Wales, Sydney, Australia, from 2018 to 2019. He has published 6 academic papers in peer-reviewed international journals. His research interests include epidemic dynamics, cybersecurity and smart grid.

Zhiguo Shi (M'10-SM'15) received the B.S. and Ph.D. degrees in electronic engineering from Zhejiang University, Hangzhou, China, in 2001 and 2006, respectively. Since 2006, he has been a Faculty Member with the Department of Information and Electronic Engineering, Zhejiang University, where he is currently a Full Professor. From 2011 to 2013, he visited the Broadband Communications Research Group, University of Waterloo, Waterloo, ON, Canada. His current research interests include signal and data processing.

Prof. Shi serves as an Editor for the IEEE Network, IET Communications, Journal of The Franklin Institute. He was a recipient of the Best Paper Award from the ISAP 2020, IEEE GLOBECOM 2019, IEEE iThings 2019, EIA MILCOM 2018, IEEE WCNC 2017, CSPS 2017, IEEE/CIC ICCC 2013, IEEE WCNC 2013, IEEE WCSP 2012. He was also a recipient of first prize for Wu Wenjun AI Science and Technology Award in 2020, first prize for Science and technology Development of the Ministry of Education in 2015 and second prize for the Scientific and Technological Award of Zhejiang Province in 2012.