Intent-driven Closed-Loop Control and Management Framework for 6G Open RAN

Jingwen Zhang*, Chungang Yang*, Ru Dong*, Yao Wang*, Alagan Anpalagan†, Qiang Ni‡, Mohsen Guizani§

Abstract—**Future mobile networks should provide on-demand services for various industries and applications with the stringent guarantees of quality of experience (QoE), which highly challenge the flexibility of network management. However, the diverse requirements of QoE and the management of heterogeneous networks create significant pressure towards communication service providers (CSPs). In the 6th generation mobile networks, the CSPs should guarantee resilient performance for the communication service consumers with less human involvement. In this work, we turn to intent-driven network and on-demand slice management, and to decrease the complexity and cost in full life cycle slice management, we first present an intent-driven closed-loop (CL) control and management framework that automates the deployment of network slices and manages resources intelligently based on the extended CL architecture. And then, we explore and exploit the deep reinforcement learning algorithm to address the problem of resource allocation, which is formulated as a Markov decision process. Finally, we demonstrate the feasibility of the proposed framework by deploying the open radio access network (RAN) infrastructure in OpenAirInterface platform and realizing the CL control and management with near real-time RAN intelligent controller. The emulation results demonstrate the effectiveness of slicing performance, measured in terms of delay and rate.**

Index Terms—**6G, Closed-loop control and management, intent-driven network, open RAN, Reinforcement learning.**

I. Introduction

THE 6th generation (6G) mobile communication system is expected to achieve ubiquitous full-band coverage, global wireless access, massive application scenarios, and on-demanding performance guarantees [1]. Meanwhile, the 6G will be compatible with typical communication scenarios such as enhanced mobile broadband, high reliability and low latency in the 5th generation (5G) mobile communication system, while also incorporating new application scenarios to provide better service to communication service consumers (CSCs) [2]. The growing demands for new and better services place a heavy burden on communication service providers (CSPs). In complex mobile communication nodes, CSPs not only need to meet consumers' resource demands but also consider system performance such as service delay and transmission quality [3]. However, it is hard to effectively monitor the real-time network status and make adjustments to resource allocation, and schedule schemes, etc., which reduce the quality of service (QoS) and the quality of experience (QoE) of CSCs. In addition, heterogeneous networks are difficult to achieve effective interconnection and global-oriented resource optimization, resulting in wasted network resources [4-5]. Thus, there call for flexibility, agility, and resilience toward the network management and control.

As an important component of end-to-end mobile network, the radio access network (RAN) also should be with programmability, flexibility and efficiency, due to the high performance vision of the 6G [6]. The open RAN alliance introduces an open RAN architecture which separates and virtualizes the conventional RAN with novel defined interfaces. For example, it was separated into a disaggregation units of the BBU, an open RAN data units (O-DU) and an open RAN radio unit (O-RU) [7], which converts the RAN to become open and fully interoperable. Due to the splitting of baseband processing, open RAN components can be flexibly placed in separate locations. Currently, the software and standards for open RAN is not yet mature. For some small-scale CSPs, the management to open-RAN may present a technological barrier due to the need for greater knowledge. Therefore, it is necessary to design and realize more automated management systems based on the open-RAN.

The 5G innovatively introduces the concepts of network slicing and on-demand services, which reduce the cost of opening private networks while saving effective network resources [8]. However, the current communication service scenarios and management methods of the 5G are insufficient to provide massive application scenarios, and the performance cannot meet the QoS requirements of various application scenarios such as industrial automation and intelligent transportation, hindering the development of emerging services in vertical industries. For example, as a typical application scenario for 5G mobile communication systems, ultra-reliable low-latency communication (URLLC) should be able to achieve end-to-end latency within 60 ms and 99.9999% reliability . In contrast, industrial smart manufacturing applications in the 6G era require end-to-end latency of 1 ms and reliability of up to 99.9999%. Considering the strengths and weaknesses of current slicing technologies, network slicing may still be inherited in 6G as a key enabling technology [9]. In order to enable CSPs to provide satisfactory services to CSCs and save overhead as much as possible, new design and technology breakthroughs in network slicing are needed. The

*Jingwen Zhang, Chungang Yang, Ru Dong, Yao Wang are with the State Key Laboratory on Integrated Services Networks, Xidian University, Xi'an, 710071, China (emails: a867689478@163.com; chgyang2010@163.com; 18302999373@163.com; yaow0518@gmail.com)

†A. Anpalagan, ECBE Department, Toronto Metropolitan University, Canada. (email: alagan@ee.torontomu.ca)

‡Qiang Ni is with the School of Computing and Communications, Lancaster University, Lancaster, LA1 4WA, UK. (email: q.ni@lancaster.ac.uk)

§M. Guizani is with Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), UAE. (email: mguizani@ieee.org)

This work was supported by the National Key Research and Development Program of China (2020YFB1807700).

novel challenges for highly flexible network management and control to simultaneously meet diverse service requirements are as follows:

- It is challenging for the conventional network management approach to obtain the real-time dynamic status of all managed network resources, leading to a heavy reliance on manual intervention and compromised accuracy.
- To meet the diverse service intents of CSCs, the CSPs require a lot of a priori knowledge. Furthermore, they must possess the capability to personalize their services according to the preferences of CSCs. These requirements increase the difficulties in management and operations for CSPs.
- Large-scale and expert-based pre-configuration makes the network error-prone, requiring human judgment of the cause of errors after problems arising and a complex error correction process. The future of mobile communication networks involves a highly dynamic environment with widespread connectivity, ultra-dense access points, and broad coverage. Static control mechanisms are likely to be inapplicable or not as effective as expected.

Hence, there is a urgent need for an improved solution that automates network management to reduce the management complexity. Intent-driven network (IDN) is a closed-loop (CL) architecture based on customers' intents to build and perform related network operations [10], and it is considered an efficient approach for automating the management of network slicing [11]. As early as 2015, the Northern Interface Working Group of Open Network Fund released the white paper intent-based network [12-13]. Then Gartner, Cisco and Huawei also put forward the definition of IDN [14-15]. In their opinion, IDN is an effective way to achieve greater automation and simplify network management. Through the intent translation and the intent verification in IDN, the service intent of CSC can be converted to the reliable network policy automatically. And the policy will be delegated for deployment to the respective components. This reduces management complexity and costs and increases network efficiency.

The ETSI established the zero-touch network and service management (ZSM) industry specification group (ZSM ISG) in 2017. To address the identified needs and challenges [16], the ISG proposes a scalable, resilient, and reliable reference architecture for end-to-end network and service automation [17], where cross-domain adaptive CL, artificial intelligence (AI) and machine learning (ML) are essential to fully automatous end-to-end management operations [18]. Adaptive CLs aim to eliminate the human interference and create a self-monitoring, self-assessment, and self-managing system. However, traditional policy-based CL solutions lack the flexibility to make quick and correct decisions in the face of highly complex networks. Combining CL with IDN, coupled with AI and ML technologies, could make network management much smarter and simpler.

Considering the strengths and weaknesses of open RAN and some of the benefits of smart technologies mentioned earlier, in order to realize the vision of millisecond service, adaptive management, and flexible architecture, we need to study the intent-driven management framework for 6G open RAN, and realize the combination of CL control and artificial intelligence. The main contributions of this work are summarized as follows:

- We propose an intent-driven closed-loop control and management framework for 6G open RAN. In this framework, we adopt network slicing and intent translation technology to meet variety QoS to respond to the diverse needs of CSCs in open RAN.
- We introduce an intent goal decompositon approach to bridge the gap between CSPs and CSCs, while simplifying the control and management of networks for CSPs. And we introduce event calculus (EC) to model the intent goals decompositon and complete the logical reasoning of each element in the model by python knowledge engine (Pyke) .
- We implement a dynamic resource management in network slices using the deep Q network (DQN) algorithm to solve MDP relying on extended CL architecture, which guarantees the performance of each communication service.

The remainder of this paper is structured as follows: Section II introduces the related work. Section III presents the system framework and also outlines each module and the workflow of our framework. Section IV presents a mathematical model based on EC. Section V models the resource management problem and introduces our algorithm to solve the problem. Section VI provides our experimental implementation and results. Finally, the conclusion is given in Section VII.

II. Related Work

A. IDN for Network Management

Standards organizations such as 3GPP, ETSI, and IETF are actively developing drafts or standards related to intent-driven networks. The concept of intent-driven managed services was proposed by 3GPP SA5, defining three holders of intents, communication service customers (CSCs), communication service providers (CSPs), and network operators (NOPs) [19]. Where the CSC represents the intent in the service, he does not need to know enough about the network. The CSP represents those intents to deploy, build, and maintain communication services. The NOP represents those intents to deploy, build, and maintain the network and underlying infrastructure, and provides various intent standard interfaces for users with distinct functionalities. The ITU-T has likewise established the FG-ML5G working group to study future networks, including 5G, using machine learning. The SUPA working group of the Internet Engineering Task Force (IETF) has defined intent as a user's abstract, high-level policy for controlling the network in autonomous networks, which lays the foundation for the future development of intent. The NMRG working group, who defines intent with "an abstract, high-level policy for operating the network" [20]. IDN, as a novel network paradigm, can automatically transform and configure network resources based on the user's intent, offering fresh insights to address the shortcomings of excessive human involvement in network management.

In [21], an overarching intent framework for managing 5G network slicing was presented. This framework was proposed to provide a simplified service platform to both users and operators. The main objective is to speed up the deployment of service requirements. In addition, the framework includes a feedback mechanism on the quality of network services. However, the language used by the user in the description is cumbersome, such as specify the sensor device, guarantee bandwidth. The proposed approach by the authors in [22] leverages intent to automate end-to-end slice configuration and orchestrate resources. Nonetheless, there is room for improvement in the network slice management functionality. Additionally, the model's ability to enhance QoS remains unclear as there is no evidence or demonstrations of service performance. In [23], the authors proposed the Intent-based Cloud Service Management (ICSM) framework. This framework automates the decision-making process for cloud service providers by taking into consideration the service requirements of cloud users. An ICSM proof-of-concept is also implemented in a laboratory to appraise the architecture performance. The authors in [24] proposed an intent-driven end-to-end network slice management architecture that can manage end-to-end network slices deployed across 5G/Pre-6G infrastructures. And it is validated in Industry 4.0 horizontal and vertical use cases. The results demonstrate the effectiveness of the framework by showing that the management architecture can deploy a network slice for over 8096 nodes in less than half a second. The authors in [25] proposed two practical heuristic algorithms to effectively solve the optimization problem in the scenario of user mobility. In [26], the authors utilized machine learning (ML) techniques to automate the configuration and operational control of 5G testbeds. Their work resulted in an intent-driven network auto-configuration and management platform that allows users to control the network by providing high-level information. The platform generates network configuration policies without the need for expert experience support automatically. In addition, the system continuously monitors the runtime performance and predicts the future CPU utilization of each virtual network function (VNF) using the monitoring information. It also makes configuration changes when necessary to avoid network errors.

The existing works on intent-driven network management have made significant contributions towards automating network management and enhancing user experience. However, some of them have cumbersome language requirements for users to specify their service requirements, while others lack continuity of service performance maintance. Addtionally, some of these approaches rely on a lot of expert experience. Therefore, further research is needed to optimize these approaches for simplifying user access and sustaining high performance.

B. CLs for Network Management

The ETSI ZSM working group [27] suggested an authorized framework for end-to-end network and automated service as shown in Fig. 1. In the zero-touch service management (ZSM) architecture they provided, there are multiple management domains (MDs) and a multi-tiered closed-loops (CLs), where lower-level CLs are responsible for self-healing within their domain, and the upper-level will coordinate and optimize all scope to manage the end-to-end network. CLs can coordinate adaptively with each other to provide a satisfactory user experience. This architecture lacks details on implementation and deployment at present. Additionally, the managed entities pertain to complex mobile networks, resulting in multidimensional and complex monitoring data, which poses challenges in drawing accurate conclusions during the analysis stage.

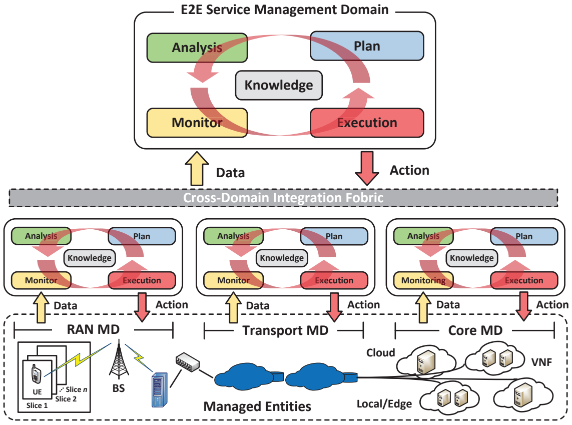


Fig. 1. The framework proposed by the ETSI ZSM [26].

The authors in [28] proposed a zero-touch service management model incorporating exposed CL. It can be applied to automate assurance service. And the model was validated by building a testbed to collect data from OSM monitoring modules and external tools. The analysis of the results shows the value of customer-driven management in the exposed CL. The authors in [27] used intent as a communication mechanism to communicate messages between different layers and MDs in the management system of the ZSM framework, allowing for a higher level of autonomy and flexibility. An intent specification based on uRLLC services is demonstrated, which specifies all goals for URLLC services according to the NEST standard, including specific technical domain aspects such as transmission latency and guaranteed bandwidth to support network slicing requirements from different vertical users. The second is intended to be used in the article to represent targets at different levels throughout the adaptive system, such as service, service, or resource levels. They are processed accordingly by CLs in the management domain at each level and decisions are made by analyzing the gap between observed entities'state and the goal obtained through the intent. The eventual goal is then to reduce the difference between the desired and the actual state. The authors in [29] proposed a zero-touch network management framework based on CLs, which is a policy-driven network management system capable of automating network management in a hierarchical multi-domain environment architecture based on ZSM through a collaborative distributed CL as well as an artificial intelligence approach. This approach reduces the complexity of network and service management and enables service assurance through the implementation of adaptive features.

The authors in [30] investigated the automation of deployment of cloud RAN systems and propose a ZSM model that enables automated resource awareness and automated RAN unit life cycle management, including instantiation, configuration, monitoring and removal. The implementation of the system provides an NMS with a graphical interface, which is convenient to operate and manage the cloud system. Performance results show that the proposed ZTC system takes only 32 seconds for the automatic deployment of a fully cloud-based RAN system without any human intervention, verifying the availability of end-to-end services.

The authors in [31] proposed a system that can receive multiple intents from human operator, and also predict the influence of every action required in the intents to detect the run-time conflicts then reslove them. This solution instantiate a CL for each intent's expected KPI. In each CL, the system dynamically considers the global state of the network to detect and reslove conflicts, and it realizes a higher level of automation.

The authors in [32] discussed the end-to-end network slicing of uRLLC with non-standalone 5G standard. Through the mobile edge computing, they demonstrated that their scheme can increase the downlink rate and reduce the end to end delay for uRLLC slice in the scenarios of enhanced mobile broadband (eMBB) and uRLLC slices. Their delay of uRLLC slice was 25 ms. The authors in [33] proposed the ai-greedy algorithm to reduce end to end delay in wireless network. The algorithm was created in an adaptive model, and it will rearrange the network through measuring the distance at every iterations. The result shown that the delay of end to end data transmission was 61.22 ms. The authors in [34] proposed a delay optimal resource allocation model between network slice and substrate network. And used uses the branch and bound method to solve it. The result shown that the end to end delay of their uRLLC slice was 30 ms. The authors in [35] proposed the service orchestration model SliceNet and implemented the end to end cognitive network slicing and slice management framework. Through the edge computing, they shorten the delay of end to end slices to provide better service in the medical scenario. And the average round-trip time from client to edge in their case was 50.68 ms. These data show that the delay is still not up to the 5G standard and further optimization is needed.

In summary, most of works in intent-driven network management is still in the management of users' input, without considering the continuous monitoring and operation of network services. Besides, the existing zero-touch network management architecture mostly toward single management domains, and the implementation of each component is not yet clear. Due to the rapid changes and high complexity of the next-generation mobile network, it is necessary to further study the whole life cycle automatic management of communication services for future network. Compared with these existing works of network management, the novel contributions of this article are twofold. On the one hand, the intent-driven closed-loop control and management framework based on open RAN. We propose that it can quickly refine and decompose the service requirements of CSCs, find and issue adaptive rules to the managed-entities. For CSPs, this eliminates the need for their extensive a priori knowledge and reduces labor costs. On the other hand, with the help of AI in the corresponding modules, the system is able to ensure the QoS such as low delay and high data rates for CSCs.

III. Intent-Driven Control and Management Framework for Open RAN

A. Intent-Driven Control and Management Framework

Based on the open RAN architecture [7] and near real-time RAN intelligent controller (RIC), we present an intent-driven control and management framework for open RAN to provide communication resources to the end-devices in scenarios for CSCs requests. An overview of the framework is shown in Fig. 2. There are three layers in this framework, including the intent layer, control and management layer, and infrastructure layer. The characteristics of each layer and the issues to be addressed are summarized as follows:

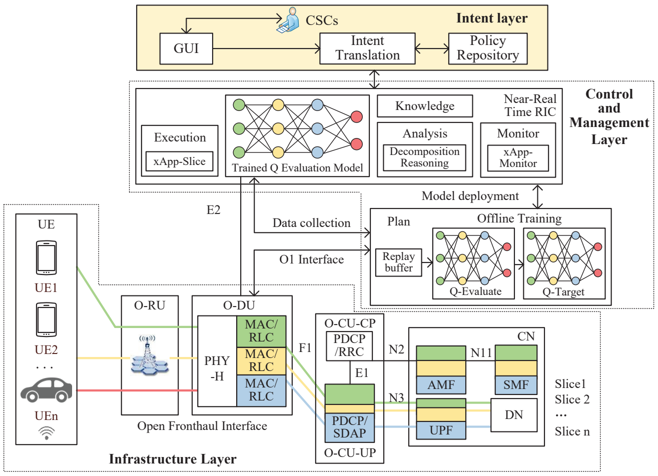


Fig. 2. Intent-driven control and management framework.

The intent layer regards the intent translation as key is accountable for receiving high-level service requirements intent from CSCs through the GUI. And converting requirements into a quintuple with the help of the policy repository.

The control and management layer, consists of a near real-time RIC and offline training model. In order to realize near-real time CL control and management on infrastructure network, we regarded MAPE-K (monitoring, analysis, plan, execution, knowledge) [36] as a valid logical control and management method. And develop the slice application (xApp-Slice), the monitor application (xApp-Monitor) in the near-real time RIC to realize corresponding modules in MAPE-K.

The infrastructure layer constructs on the disaggregated open RAN which composed of specific elements that will be presented subsequently. And the network slices are logically deployed on the infrastructure network.

B. Disaggregated Open RAN

The core of open RAN's concept is the separation of software from hardware and vendors. This concept was first introduced by 3GPP in Release 14 [37]. The existing baseband unit (BBU) is broken down into three components, which are radio unit (RU), distribution unit (DU) and central unit (CU). And multiple RUs can be connected to a single DU to reduce the cost of baseband resources by centralized processing multiple need. Moreover, in open RAN, these units link to RAN intelligent controller with open standardized interfaces to control and manage the network in near real-time CL.

E2-Interface: E2 is a logical interface connecting the near-real time RAN intelligent controller (near-RT RIC) with the underlying open RAN nodes. Due to the limitations of radio access technology (RAT), the near-RT RIC function which is bundled with O-CU may only control the underlying open RAN nodes (E2 Nodes) O-CU and O-DU.

O1-Interface: The O1 interface is between open RAN managed element and the management entity. This interface is advantageous to the non-real time (non-RT) control of RAN elements and resources. And the offline training model in non-RT RIC can use the O1 interface to acquire data.

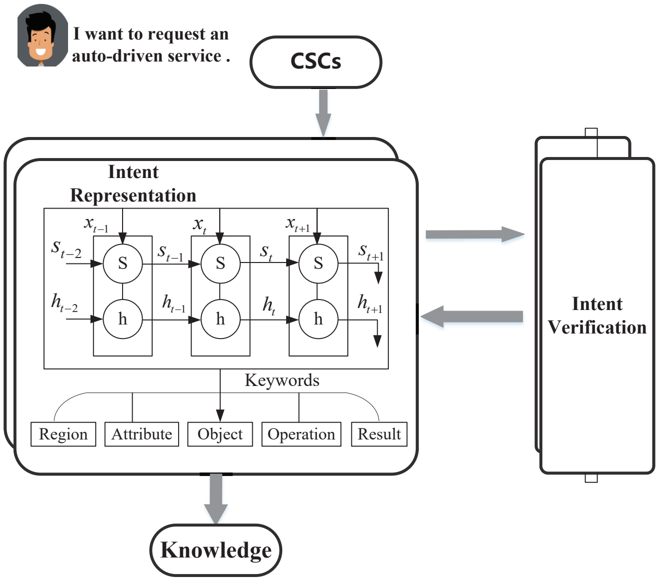


Fig. 3. Intent translation.

Open Fronthaul Interface: The Open Fronthaul interface allows DUs manage the multiple O-RUs. It is used for performance reporting or for initializing and configuring operating parameters. And it is responsible for synchronizing the time between the O-DU and the O-RU.

As shown in Fig. 2, the logical node O-RU consists of redio frequency (RF) and low physical (PHY-L). The O-DU is composed of high physical (PHY-H), Medium Access Control (MAC), and Radio Link Control (RLC). The O-CU is divided into two parts: the user plane contains packet data convergence protocol (PDCP) and the service data adaption protocol (SDAP), and the control plane specifically includes the radio resource control (RRC) to coordinate the user plane. Every possible slice all include the network function units MAC, RLC, PDCP, SDAP to ensure normal communication service.

C. Intent Translation

In our framework, the intent translation as a key technology is responsible for parsing the input intent goal into the form of a quintuple and verifying the accuracy of the translation, and then storing the correct translation results in the knowledge base. Both the analysis and plan stages will use this knowledge to appropriate resource management rules. This process is implemented through two modules, Intent Representation and Intent Verification, as shown in Fig. 3.

In the application process of the framework, CSCs first express the intent goal in the form of natural language, which is ambiguous and does not follow a specific form. Therefore, the representation module of intent translation needs to translate the intent expressed in natural language by Bi-directional Long Short-Term Memory and morphological rules, and output the understandable and regularized intent goal. Secondly, the intent verification module will analyze the accuracy and completeness of the characterized intent of the transcribed intent, and then continuously monitors possible conflicts. The intent translation module is the first step to realize the intent-driven network management.

The intent verification is a necessary consideration in order for the intend goal to be executed accurately. The intent verification is based on the commitment theory, which guarantees that the transcreation result can satisfy the original goal. In case of multiple intent conflicts, the conflict deconfliction is performed and the conflict-free intent quintet is distributed to the knowledge of MAPE-K.

D. Decomposition and Reasoning

Through the intent translation, we can get the formalized intent goals of quintuples from the knowledge. Treating each intent goal as an event, we apply the EC to model the refinement processes of intent goals. And the Pyke is a toolkit we used to implement the rules and formulas of EC. It brings in a form of logic programming to the python inspired by Prolog. Thus, it acts as the reasoner of EC and is also associated with the offline training algorithm model to assist in the selective collection of algorithmic information and the setting of decision spaces. It acts as the analysis stage of MAPE-K. And the decomposition model of EC will be introduced later as a separate section.

E. Offline Training Algorithm

The Offline training algorithm is an important component of the control and management layer. It can make different decisions under different resource environments. In our framework, we adopt the DQN as the main body of the algorithm. And it can reduce the action space in complex environments with the help of decomposition and reasoning. The algorithm allows for greater flexibility and adaptability in managing complex and dynamic network environments, enabling the CSPs to optimize network performance, improve resource utilization, and ensure efficient communication resource allocation more simply. It acts as the plan stage of MAPE-K.

F. XApp-Slice and XApp-Monitor in Near Real Time RIC

In the open RAN framework, the near-RT RIC is responsible for the real-time control of the RAN elements and resources by collecting and manipulating fine-grained data on the E2 interface. In our framework, we enhanced the function of the near-RT RIC by creating new xApps to deploy network slicing and monitor the RAN in real time using FlexRIC, a flexible and efficient software development kit (SDK) [38]. The E2 agent has access to internal RAN components within the infrastructure stack to monitor and modify the RAN parameters. The xApp-Slice can create slices, bind terminals to these slices or delete them. Meanwhile, the trained model enables the xApp-slice to dynamically obtain management operations for each slice and send them to the corresponding resources via the E2 interface. It acts as the execution stage of MAPE-K. At regular intervals, the trained model will be replaced by the offline training model. Through the new xApps, logical slices with slices' label, base-station's id and users list can be deployed on the open RAN. By utilizing network slicing, CSPs can accommodate the diversified requirements imposed by the verticals. Network slices with different QoS criteria can service for different UEs on demand, such as URLLC services. RAN slicing plays an important role in end-to-end network slicing, ensuring that every service shares communication resources efficiently.

G. Extended Closed-Loop Management

CL is an existing management method [16] in which the system is located on managed entities with specific goals. It is able to monitor and act on the managed entity by analysis, plan, and the execution cycle entity. The CL is achieved through the interaction of management services, which are mainly provided by functional modules in its different phases. The traditional MAPE-K CL consists of four stages: monitor, analysis, plan, and execution, and we have integrated the reward and state services from our DQN model in the traditional architecture, and defined in detail what exactly needs to be stored in the knowledge base, which will be detailed later. Based on the extended CL architecture logically, we developed each model in near-RT RIC to implement the resource manangement for open RAN.

Specifically, in the extended CL architecture we proposed, and the knowledge of this CL consists of an environment, a system, a target, and an adaptive rules. The environment contains abstract information of the relevant state of the managed entities at present. And the system model mainly contains information such as nodes topology. The goal model contains sub-goals that describe the intent of the actual business application of the management system. The sub-goals can be abstract goals or operational goals. The goal model can determine whether the current state satisfies the sub-goals. The key to the knowledge is the adaptation rule model. The self-adaptation rules consist of three parts: preconditions, data manipulation commands and postconditions. The preconditions describe when the adaptation rules are actually applicable and need to be evaluated on the basis of environment and system. The data manipulation commands describe the change of the control data when the rule is applied. The postconditions describe how the environment should be affected after the action is executed in the ideal state and whether it meets the system intent goals. The monitor stage tracks the state and behavior of all nodes for open RAN to collect information about resources during the entire service provision period, and the information is used to update the system models and the environment models. The analysis stage is where these models are used to decide if managed entities adjustments are needed. The decision is depended on the gap between the system and the intent goal. If SLA violations exceeding fault tolerance values, adaptive rules must be applied to approach the intent again. If a reliable management rule can be found, it will be employed to the system through issuing the analogous configuration parameters within the execution agent. In addition, the employed rules and their expected effects are writed in knowledge base for subsequent queries. If not, the system needs to wait the CL for rule learning.

H. Intent-Driven Closed-Loop Control and Management

After analyzing the characteristics and key technologies of the three layers above, the process of our proposed management framework is shown in Fig. 4. First, the CSCs use the interactive interface to input the web-level business intent. This intent is the goal to be achieved and requires the user to inform the life cycle of the system requirement service. Then, the intent manager completes the semantic extraction based on the CSCs' input and identifies the corresponding information from the knowledge base. It also completes intent decomposition, and gets the services and the SLA that the system will provided. Next, the manager provides the results to the module corresponding to the analysis phase in MAPE-K. If the gap between the environment and the intent goal is greater than the fault tolerance value, corresponding adjustment is needed. The relevant information is then informed to the plan stage. In this phase, it queries the knowledge base for rules that can be applied to adjust the service in the current state. If a reliable rule can be found, it is applied to the managed entities through publishing the analogous operation and configuration in the execution phase. If no, a new assignment rule is learned through DQN. The new rule is then stored in the knowledge base. The monitor provides real-time monitoring of the relevant open RAN resources throughout the service lifecycle. And in our framework, the analysis stage is realized by the decomposition reasoning. The main of plan stage is realized by offline learning in non-RT RIC. The execution is mainly achieved through the E2 Agent and xApp-Slice. And the xApp for monitoring completes the resources information collection in real-time.

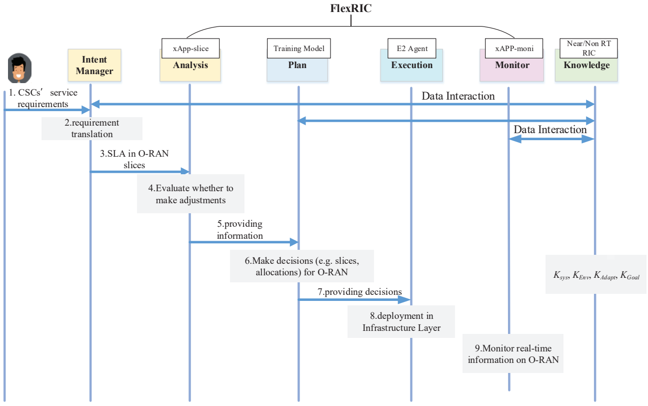

FlexRIC
Intent Manager
xApp-slice
Analysis
Training Model
Plan
E2 Agent
Execution
xAPP-moni
Monitor
Near/Non RT RIC
Knowledge
1. CSCs′ service requirements
Data Interaction
2.requirement translation
Data Interaction
3.SLA in O-RAN slices
4.Evaluate whether to make adjustments
5.providing information
6.Make decisions (e.g. slices, allocations) for O-RAN
K_{sys}, K_{Env}, K_{Adapt}, K_{Goal}
7.providing decisions
8.deployment in Infrastructure Layer
9.Monitor real-time information on O-RAN

Fig. 4. Process of open RAN's CL control and management.

IV. Intent Decomposition Model

A coarse-grained abstract intent goal can be refined into multiple fine-grained sub-intent goals. This section presents a modeling approach and process for decomposing abstract intents into multiple levels. The top of the model is the high-level abstract intent goals derived from the consumers. And the bottom layer of the model can be understood as the most relevant solution. For CSPs, after receiving the high-level intent goals from CSCs. They should implement the solution to deploy the satisfactory services to consumers. For example, completing radio access network configurations including cell list to meet the intent "deploy a URLLC service at the edge of the network" from CSCs. Due to the complex network structure and the changing network status of the controlled area. It is difficult to find the most suitable solution quickly and accurately. This mathematical model bridges the gap between the CSCs requirements and CSPs services, helping them save management costs and reduce the probability of errors. Specifically, the analysis stage in our framework will decompose the CSCs' intent goal according to the facts and rules which modelled by the EC in the knowledge until the operational goals are found. All these abstract intent goals, the facts and rules are stored in the knowledge base as

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3312795

our expert experience. Through the logical reasoning, the abstract intent goals can be refined into multiple possible operational goals rapidly. During this decomposition process, all intent goals need to be firstly represented as a quintuplet ($< Region >, < Attribute >, < Object >, < Operation >$ $, < Result >$), where $< Region >$ describes the service identity; $< Attribute >$ describes the service attributes; $< Object >$ describes the object targeted by the service; $< Result >$ describes the expected state, including the link identity, link state, delay, packet loss rate, bandwidth, etc.

The environment is represented by a two-tuple (A, W), where $A = \{A_1, ..., A_i, ..., A_n\}$ denotes the functional or the non-functional attributes, and $W = \{W_1, ..., W_i, ..., W_n\}$ denotes the weight of each represented attribute.

The decomposition process is a triad (G, CL, D), where G denotes the network-level business service that the user is trying to implement or the expected outcome at the end of the service. The CL denotes the control relationships between the service activities, and D denotes the data relationships between the activities.

Based on the abstraction level, the intent goals are classified as abstract goals and actionable goals. Actionable goals correspond to concrete measures for the net elements and are located at the lowest level of the tree. Each abstract goal can be decomposed into several subgoals in the form of AND/OR, which are other abstract or actionable goals with weight constraints. Usually, the only way to find a concrete operation that matches the abstract goal is through continuous refinement.

In the goal decomposition process, we use event calculus (EC) to represent goals in a high-level abstract and apply logical reasoning to mathematically model the goal decomposition process. EC is a logic-based formalism that is mainly used to represent and reason about behaviors and their expected effects, i.e., EC can infer what is correct at a given time and action of occurrence, and its basic elements are events and fluents. Where events are anything that happen in a given definition domain at any given time that causes a partial change in the state of the definition domain. Fluents, on the other hand, are any measurable part of the domain that changes over time, and the set of the fluent values within the definition domain is the overall state of the definition domain. The two elements together are used to describe what happens over time, their effects, and ultimately how the definition domain evolves. In this model, the predicate is the most important component. It is used to specify the content of the object representation in the system or the relationship between the objects. In the EC model, the predicates that are commonly used are as follows:

- $Initiates(a, f, t)$**:** indicates that in time t, after the action a, fluents f started to be established.
- $Terminates(a, f, t)$**:** indicates that in time t, after the action a, fluents f end.
- $Initially_p(f)$**:** indicates that the fluents f holds from $t = 0$.
- $Initially_N(f)$**:** indicates that the fluents f does not holds from $t = 0$.
- $Happens(a, t_1, t_2)$**:** action a happens at $t1$ and ends at $t2$, satisfy $t1$ is less than $t2$.
- $HoldsAt(f, t)$**:** indicates that the fluents f holds from t.
- $Clipped(t_1, f, t_2)$**:** indicates that the fluents f ends up holding at some point between $t1$ and $t2$.
- $Declipped(t_1, f, t_2)$**:** indicates that the fluents f starts up holding at some point between $t1$ and $t2$.

To model the domain using the above list of predicates and interpret the results, the above predicates are transformed into the following basic axioms of EC as

$$HoldsAt(f, t) \leftarrow Initially(f) \wedge \neg Clipped(0, f, t), \tag{1}$$

$$\begin{aligned} HoldsAt\,(f, t_3) \leftarrow & \, Happen\,(\alpha, t_1, t_2) \\ & \wedge Initiates\,(\alpha, f, t_1) \wedge t_2 < t_3 \\ & \wedge \neg Clipped\,(t_1, f, t_3)\,, \end{aligned} \tag{2}$$

$$\begin{aligned} Clipped\,(t_1, f, t_2) \leftrightarrow \exists \alpha, t :& Happens\,(\alpha, t_1, t_2) \\ & \wedge t_1 < t < t_2 \wedge Terminates(\alpha, f, t). \end{aligned} \tag{3}$$

Here Axiom (1) states that if fluent f holds at $t = 0$ and fluent f does not terminate between 0–t, then fluent f holds at t. Axiom (2) states that fluent f holds at t if action a happens at t_1 and ends at t_2, at t_1, the action a makes fluent f start to hold, and it does not terminate between t_1–t_3. Then fluent f holds at t_3. Axiom (3) states that if there exists an action a occurring at t (t is some point between t_1–t_2) and fluents f is terminated at t by the occurrence of action a, then fluents f ends to hold at some point between t_1–t_2 . Based on the above axioms, it is obvious that $Initiates$ and $Terminates$ are used to describe the action of the event and express action information; the combined narrative of $Initially$ and $Happenes$ is used to define temporal information; $<$ is used to determine the temporal priority and is used in the presence of constraints; $HoldsAt$ and $Clipped$ are used as core axioms. We will explain the application of the model in detail in Section VI in conjunction with the service case.

V. Resource Management Model for Open RAN Slicing

A. Problem Model as Markov Decision Process

The efficient resource allocation methods are crucial for network management. Thus, they play a fundamental role in our knowledge base regarding as the operational goals. In the control and management layer of our framework, the resource allocation focus on effectively scheduling MAC resources in O-DU of each slice. Static resource management is flawed because of the unpredictability inherent in the wireless channel, we consider the dynamic approach to reassign resources in time depended on network and channel changes. To do that, the resource allocation among the open RAN slices is represented as a Markov decision process (MDP) in our modeling. Since there are limited communication resources shared between networks with different service slices, the goal of our MDP problem is to minimize the probability of violating the service level agreement (SLA), which identifies both the services required and the expected level of all service slices.

$$\begin{gathered} \arg\min P(|Q_{slicen}(e, u) - \lambda_{slicen}| \geq \in_n), \\ \text{s.t.}, \sum_{n=1}^{N} \sum_{m=1}^{M} e_n u_m \leq K, \end{gathered} \tag{4}$$

where λ_{slicen} indicates the desired threshold in QoS requested by slice n, and $\in_n$ indicates the tolerated error range. e and u are vector of e_n and u_m as resource allocation indicator. N indicates total number of RAN slices providing communication services. M is the amount of UEs in each slice. K indicates total communication resources available for allocation in MAC. The above variables have been summarized in Table I. MDP provides a usable mathematical framework for solving optimization problems. Therefore, we model (4)

TABLE I
MAPPING FROM DISAGGREGATED OPEN RAN ENVIRONMENT TO DQN.

Variable	Description	Content
λ_{slicen}	desired threshold in QoS requested by slice n	Limitations corresponding to the slice getting from the result of translation
$Q_{slicen}(\boldsymbol{e},\boldsymbol{u})$	performance state specific values	The actual value corresponding to the expected performance of each slice
e_n	resource allocation indicator	Number of resources allocated per slice
u_m	resource allocation indicator	Number of resources allocated per user
K	total communication resources	available resources in communication environment
$\boldsymbol{S_t} = \{s_t^1, ...s_t^i\}$	current state of each step	Existing resource distribution; Resource utilization; Throughput ; Delay ; Reliability.
$\boldsymbol{A_t} = \{a_t^1, ...a_t^j\}$	action of state at each step	The adjustment of RBs allocation;
r_t	reward at each step	$+1$ for decreasing the goal distance function; -1 for increasing the goal distance function.
$Q(\boldsymbol{S_t}, \boldsymbol{A_t})$	Q-function maps state -action to Q-values	current reward and discounted future max Q value
$\boldsymbol{\theta}$	all parameters of the neural network	containing multiple parameters, each representing a weight in the neural network
$L(\boldsymbol{\theta})$	Loss function	calculate the difference of Q-value between the Q-eval network and Q-target network

as a single-agent MDP, and solve it by DQN, an offline deep reinforcement learning algorithm. As our framework in Fig. 2, the offline algorithm collects the useful information over time by the xApp-Monitor in the near-RT RIC. It is the intelligent agent responsible for making the resource allocation plans to manage open RAN. The important members of our MDP are outlined below:

- **State Space:** The state of controller agent represents the open RAN status in each time step, includes the setting of related UEs, available RBs, and QoS of $slice_n$.
- **Action Space:** The agent needs to decide how resources should be allocated to the open RAN slices and associated UEs. Both the open RAN slices and UEs can utilize multiple RBs to satisfy the desired QoS, as a result, the action is characterized by row vectors e, u. In every step time t, the agent will decide to perform action based on policy.
- **Reward Function:** The value of reward depends on whether it reasonably allocates the required resources. An operation is considered successful if it satisfies the SLA constraint. Since our goal is to minimize the SLA violation probability, the reward can be expressed as the complement of violation probability.

B. Open RAN Slicing Based on Deep Q Network Algorithm

We use the DQN approach to investigate the defined MDP model. And it contains two essensial phases: the training and the implementation. In the training phase, two neural networks are used in an offline manner. Rewards, actions, policies, states, agents, and environments are all essential elements in the process of training. In our framework, the neural networks of Q-evaluation (Q-eval) are trained offline with the train data collected by the xApp-Monitor in a period of time. The mapping of the DQN algorithm from the OAI-based disaggregated open RAN environment to each parameter of model is shown in the Table I.

The agent retrieves the state $\boldsymbol{S_t} = \{s_t^1, s_t^2, s_t^3, s_t^4, s_t^5\}$ from the underlying network during each training iteration, where s_t^1 is the existing resource distribution, s_t^2 is the resource utilization, s_t^3 is the throughput, s_t^4 is the delay, and s_t^5 is the reliability.

Based on the algorithm model, the output to the state will be the action $\boldsymbol{A_t} = \{a_t^1, a_t^2, a_t^3\}$, where a_t^1 is the resource allocation in slice 1, a_t^2 is the resource allocation in slice 2, a_t^3 is the resource allocation in slice 3. And the immediate rewards r_t will be received. Regardless of whether the state is positive or negative, the intelligent agent must learn to maximize potential rewards through

$$R_t = \sum_{k=0}^{N} \gamma^k r_{t+k+1}, \tag{5}$$

where N refers to the total number of training steps throughout the learning process, $\gamma (0 < \gamma < 1)$ represents the discount factor represents the present value of future rewards. Equation (5) serves as a reference to predict the future reward that will be obtained. But one disadvantage is that the reward may vary randomly based on the initial value of the state. As a result, during each training step, the expected reward value for taking action focusing on a particular state is

$$\begin{aligned} Q(\boldsymbol{S_t}, \boldsymbol{A_t}) &= E[R_t[\boldsymbol{S_t}, \boldsymbol{A_t}]] \\ &= E[r_t + \gamma \mathrm{max}_{\mathrm{a_{t+1}} \subseteq \mathrm{A}} Q(\boldsymbol{S_{t+1}}, a_{t+1})], \end{aligned} \tag{6}$$

which represents action function. It is responsible for selecting the optimal control action that maximizes the reward value for each state. However, since intelligent agents cannot initially determine the optimal control action, it is crucial to explore untrained state-action pairs during the learning process. The greedy strategy $\varepsilon - greedy$, $1 - \varepsilon$ means the probability that previously stored Q values are utilized. Random actions are learned with a uniform distribution probability of ε. In DQN, actions updates are performed by

$$\begin{aligned} Q(\boldsymbol{S_t}, \boldsymbol{A_t}) &\leftarrow (1-\alpha) Q(\boldsymbol{S_t}, \boldsymbol{A_t}) \\ &+ \alpha (r_t + \gamma \mathrm{max}_{\mathrm{a_{t+1}} \subseteq \mathrm{A}} Q(\boldsymbol{S_{t+1}}, a_{t+1})), \end{aligned} \tag{7}$$

In (7), α is a constant represents the learning rate that determines the number of action updates. Thus, (7) is converted to

$$\begin{aligned} & Q(\boldsymbol{S_t}, \boldsymbol{A_t}) \leftarrow Q(\boldsymbol{S_t}, \boldsymbol{A_t}) \\ & + \alpha (r_t + \gamma \mathrm{max}_{\mathrm{a_{t+1}} \subseteq \mathrm{A}} Q(\boldsymbol{S_{t+1}}, a_{t+1}) - Q(\boldsymbol{S_t}, \boldsymbol{A_t})). \end{aligned} \tag{8}$$

If the algorithm converges, the second term in (8) will become zero. Additionally, within the parentheses, it represents the error of the temporal difference. In the algorithm, the optimal action is refined through frequent updates of the action values that this inaccuracy is approximated to be 0.

The Q-network is responsible for calculating the action-value function in DQN, and it is named so since it is approximated by the neural network. During the training process, the Q-network's parameters are updated to minimize the inaccuracy between the predicted and actual action values, which is regarded as the function that measures the loss. Thus, the Q-value of the target can be the label of the training, the loss fucntion expressed as the mean square error (MSE) of the target and the current is

$$L(\theta) = E[(y_t - Q(\boldsymbol{S_t}, \boldsymbol{A_t}; \boldsymbol{\theta}))^2], \tag{9}$$

$$y_t = r_t + \gamma \mathrm{max}_{\mathrm{a_{t+1}} \subseteq \mathrm{A}} Q(\boldsymbol{S_{t+1}}, a_{t+1}; \boldsymbol{\theta}), \tag{10}$$

where y_t denotes the Q-values of Q-target network, $\boldsymbol{\theta}$ is employed to depict all the parameters of Q-evaluate network. And the intelligent agent can acquire knowledge of action values by iteratively updating them towards the target.

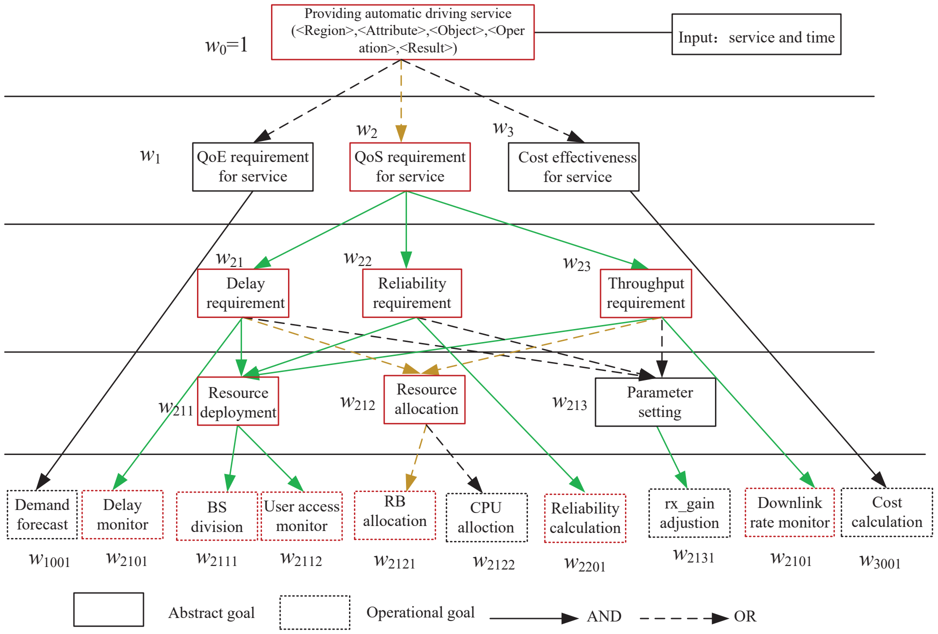


Fig. 5. Goal decomposition on CL control and management.

Learning occurs through the interaction of an intelligent agent and its environment, where pre-existing experiences are used to set states, actions, and rewards. By receiving experience inputs in a sequential manner, the agent can establish correlations between the data as it learns. In order to eliminate correlations, experience replay is necessary for the neural network. And experiences e_t, representing by the current state, the current action, the current action and next moment status, which can be stored in the pool $D_t = e_1, e_2......e_t$. To eliminate correlations during the learning process, the neural network selects small random batches of samples from the experience pool and uses them for training.

To accelerate the learning process and improve its stability, a Q-network with a fixed target is employed. The target oscillates within a specific range due to $Q(\boldsymbol{S_{t+1}}, a_{t+1}; \boldsymbol{\theta})$ in (10). Consequently, setting the target to a specific time interval ensures that target values will not oscillate as they normally would. When the parameters are bound to $\boldsymbol{\theta}'$, (10) is replaced with

$$y_t = r_t + \gamma \max_{a_{t+1} \subseteq \boldsymbol{A}} Q'(\boldsymbol{S_{t+1}}, a_{t+1}; \boldsymbol{\theta}'). \quad (11)$$

The fixed range for the target is referred to as the renew space of θ'. Utilizing it, the Q-target network Q-value in i iterations is computed, as opposed to the present Q-network's update. $Q'(s_{t+1}, a_t; \boldsymbol{\theta}')$ serves as the target for the Q-network and is updated at the intervals of steps, rather than through iterative updates over time. The current Q-network's parameters, $\boldsymbol{\theta}$, is updated through gradient backpropagation. And the gradient is equal to the loss function taking the partial derivative of $\boldsymbol{\theta}$ as

$$\nabla L(\boldsymbol{\theta}) = E[((y_t - Q(\boldsymbol{S_t}, \boldsymbol{A_t}; \boldsymbol{\theta}))\nabla \boldsymbol{\theta} Q(\boldsymbol{S_t}, \boldsymbol{A_t}; \boldsymbol{\theta})]. \quad (12)$$

The pseudo-code for the algorithm, based on the above formula and algorithmic model, has been given in our previous work [39].

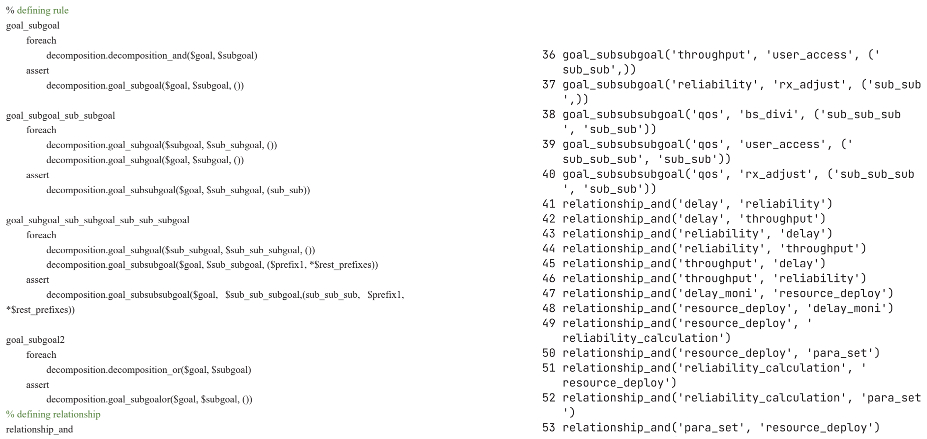
VI. An Example of Realizing Closed-Loop Control and Management in Our Framework

After introducing the key technologies and processes of the framework, we will further demonstrate how it achieves the CL control and management through a typical business scenario for the CSPs. Considering that future networks should be flexible, our work focuses on the slices in the open RAN.

A. Case Study

As a typical business scenario, automatic driving service requires extremely low delay and ultra-high rate. This implies that for CSPs, providing such services to CSCs requires a greater investment of time and effort. The authors in [40-41] using the enhanced Federated Learning to accelerate the resource scheduling for automatic driving. It illustrates the need for AI to achieve performance optimisation in automatic driving. Thus, we use it as the case to demonstrate CL control and management. Fig. 5 shows the intent goal decomposition tree of "automatic driving service" and the goal-oriented description. Through the NEST standard [27], we map the intent of the CSC to specific QoS metrics, delay, reliability and throughput. With the help of certain prior knowledge in the base, we further obtain a possible set of sub-goal. The importance of each sub-goal to the higher-level goal is represented by weight, which are obtained by training feed-forward neural network on the training set. Each abstract intent goal can be decomposed into several sub-goals in the form of AND/OR graph. In the following we will describe in detail the inference process for a particular decomposition flow of "providing automatic driving service" in Fig. 5 through the EC model. First, after the intent translation, the abstract natural language has been represented in the form of a quintuple as Table II. Since "providing automatic driving service" is the initial intent without decomposition, it will correspond to all the possible operation goals which stored in the knowledge base in advance. As the reasoner of EC, the Pyke engine will give the possible decomposition of the abstract goal. It depends

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3312795



```
% defining rule
goal_subgoal
    foreach
        decomposition.decomposition_and($goal, $subgoal)
    assert
        decomposition.goal_subgoal($goal, $subgoal, ())

goal_subgoal_sub_subgoal
    foreach
        decomposition.goal_subgoal($subgoal, $sub_subgoal, ())
        decomposition.goal_subgoal($goal, $subgoal, ())
    assert
        decomposition.goal_subsubgoal($goal, $sub_subgoal, (sub_sub))

goal_subgoal_sub_subgoal_sub_sub_subgoal
    foreach
        decomposition.goal_subgoal($sub_subgoal, $sub_sub_subgoal, ())
        decomposition.goal_subsubgoal($goal, $sub_subgoal, ($prefix1, *$rest_prefixes))
    assert
        decomposition.goal_subsubsubgoal($goal,   $sub_sub_subgoal,(sub_sub_sub,   $prefix1,
*$rest_prefixes))

goal_subgoal2
    foreach
        decomposition.decomposition_or($goal, $subgoal)
    assert
        decomposition.goal_subgoalor($goal, $subgoal, ())
% defining relationship
relationship_and
```

(a) $decomposition\_knowledge.krb$.

```
goal_subsubgoal('throughput', 'user_access', ('
sub_sub',))
goal_subsubgoal('reliability', 'rx_adjust', ('sub_sub
',))
goal_subsubsubgoal('qos', 'bs_divi', ('sub_sub_sub
', 'sub_sub'))
goal_subsubsubgoal('qos', 'user_access', ('
sub_sub_sub', 'sub_sub'))
goal_subsubsubgoal('qos', 'rx_adjust', ('sub_sub_sub
', 'sub_sub'))
relationship_and('delay', 'reliability')
relationship_and('delay', 'throughput')
relationship_and('reliability', 'delay')
relationship_and('reliability', 'throughput')
relationship_and('throughput', 'delay')
relationship_and('throughput', 'reliability')
relationship_and('delay_moni', 'resource_deploy')
relationship_and('resource_deploy', 'delay_moni')
relationship_and('resource_deploy', '
reliability_calculation')
relationship_and('resource_deploy', 'para_set')
relationship_and('reliability_calculation', '
resource_deploy')
relationship_and('reliability_calculation', 'para_set
')
relationship_and('para_set', 'resource_deploy')
```

(b) Reasoning result.

Fig. 6. The implement of goal decomposition.

TABLE II
INTENT TRANSLATION.

Intent	**providing automatic driving service**
Region	mobile network
Attribute	abstract goal
Object	managed-entites
Operation	demand forecast; delay monitor ; rx _ gain adjustion; downlink rate monitor ...
Result	delay $<$ 20ms; rate $>$ 1Mbps

on the pre-defined axioms implemented in the EC. In order to complete the decomposition of the abstract goal until the element corresponding to the attribute that is an operation goal is found, some relevant axioms need to be known in advance.

The resource allocation $allocation$ represents a complex action which is compounded by the atomic actions RB allocation $RBadjust$, CPU allocation CPU as

$$\begin{aligned} &Happens(allocation, t_1, t_2) \leftarrow \\ &Happens\left(RBadjust, t_1, t_2\right) \vee \\ &Happens\left(CPU, t_1, t_2\right). \end{aligned} \tag{13}$$

Then, A fluent—end-to-end delay requirement $delay$ reflects the network real-time state of the uRLLC service provided by the CSPs. This state is affected by the complex action $allocation$ but the effect requires further learning and is temporarily represented by unknown. That is to say that the delay constraint cannot be inferred to be satisfied when the resource allocation is true as

$$\begin{aligned} &Initiates(allocation \wedge delaymon, delay, t) \leftarrow \\ &Happens(allocation, t_1, t_2) \wedge t_1 < t < t_2 \wedge \\ &Happens(delaymon, t_1, t_2) \leftarrow \\ &[Happens(RBadjust,\ t_1, t_2) \vee \\ &Happens(CPU,\ t_1, t_2)] \wedge \\ &Happens(delaymon, , t_1, t_2). \end{aligned} \tag{14}$$

A fluent reliability requirement $reliability$ reflects the degree of accuracy of the provided service and is used to measure how often the overall adaptive management system makes the correct decision, which can be calculated by the mean squared error (MSE). Since unsupervised learning has no historical data, it can be solved by comparing the actual target distance after performing the action with the neural network prediction; A fluent—throughput requirement $throughput$ mainly reflects the downlink rate when the data is download; A fluent—provides satisfactory automatic driving service is true only when the requirements of $delay$, $reliability$, $throughput$ are both satisfied as

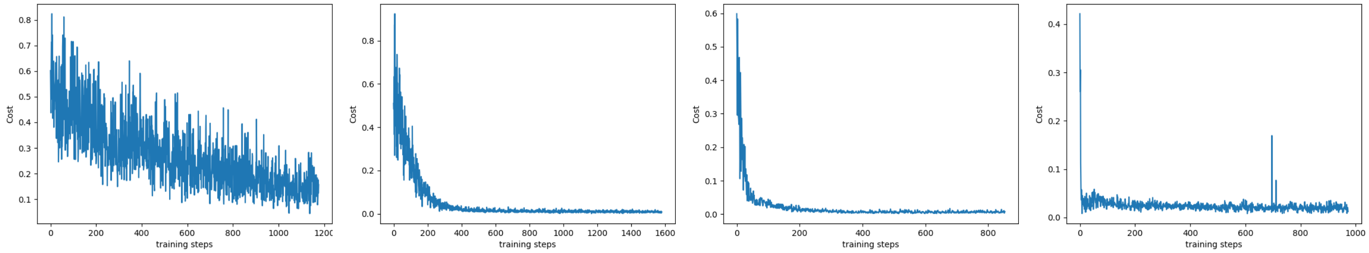
$$\begin{aligned} &HoldsAt(self-driven, t) \leftarrow Initially_P(delay) \wedge \neg \\ &Clipped(0, delay, t) \wedge Initially_P(reliability) \wedge \neg \\ &Clipped(0, reliability, t) \wedge Initially_P(throughput) \wedge \neg \\ &Clipped(0, throughput, t). \end{aligned} \tag{15}$$

B. Simulation Setup and Environment Details for Open RAN

In order to illustrate the control and management process of our framework, we first simulate our managed-entities with the help of the OAI platform. The RU handles each part of the digital front end (DFE) and PHY layer as well as the digital beamforming functions. The DU runs the RLC, the MAC and part of the PHY layer. And the CU is responsible for its operation. The RU is deployed on processors Intel Core i7-1260 CPU@2.10GHz, along with usrp b210. And the controller's operating system is Ubuntu 18.04 with low-latency kernel "Linux 4.8.0-52". The evolved packet core (EPC) uses VMware Workstation Pro 16 to deploy in i5-3210M CPU, indicating that the OAI has implemented all the functionalities of the EPC. What's more, the PC deploying the RU runs FlexRIC and xApps as the SD-RAN controller, the UEs in the network model are deployed by smart phone—Redmi 4A and VMware. They are linked to the base station through antenna and bridge. The decomposition of the goals implemented through logical reasoning by Pyke 1.1.1. DQN algorithm is designed on Python 3.6 platform, using TensorFlow 1.13.1 to build the Q-eval network and Q-target network for the offline training, and it runs on Ubuntu 18.04. The relevant settings of the test bed can be clearly seen in Table III.

C. Analysis of Results

The above events, fluent, axioms have been translated into the programmes of Pyke, and the programmes $decomposition\_knowledge.krb$ as shown in Fig. 6(a) have



(a) $\alpha = 0.0001$.

(b) $\alpha = 0.001$.

(c) $\alpha = 0.01$.

(d) $\alpha = 0.1$.

Fig. 7. Impact of learning rate on loss function.

TABLE III
SIMULATION SETTING.

Role	**Hardware/Software**
the RU	Intel Core i7-1260 CPU@2.10GHz usrp b210
the Core	i5-3210M CPU
the UEs	Redmi 4A and VMware
the operating system	Ubuntu 18.0 “Linux 4.8.0-52”
logical reasoning	Pyke 1.1.1
algorithm	Python 3.6
neural network	TensorFlow 1.13.1
Parameter name	**Parameter setting**
epsilon	increasing from 0 to 0.9
episode	200
learning rate	0.001
discount factor	0.9
batch size	64
memory size	5000
maximum steps of each episode	1000

TABLE IV
GOAL DECOMPOSITION AND REASONING.

Intent	**RB allocation**
Region	mobile network
Attribute	operationl goal
Object	O-DU
Operation	scheduling free RBs for new slice; reduce the allocation for new slice
Result	delay $<$ 20ms; rate $>$ 1Mbps

been stored in the knowledge base. Through the defining the different types of facts, rules, the backend algorithm in the plan stage can obtain the operational goals to reduce the decision space. And the reasoning result is shown in Fig. 6(b).

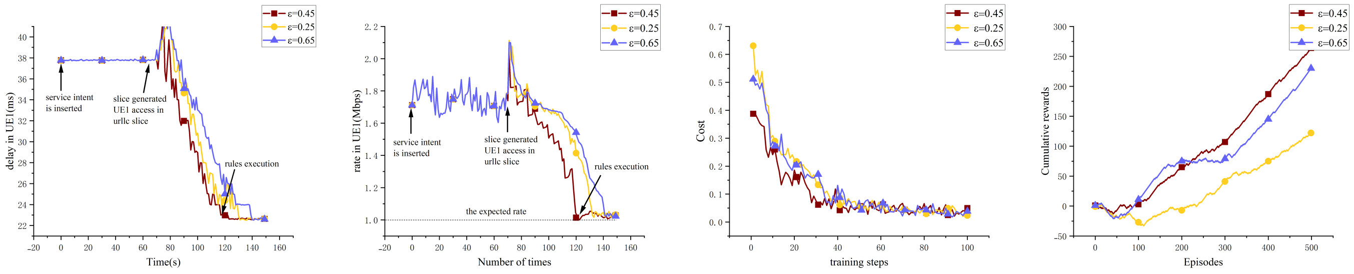
To demonstrate the feasibility of our framework, we presented the process of CL control and management for open RAN, which can provide satisfactory services to CSCs in automatic driving instance. First, the management system will translate the highest level intent goal provided by the CSCs as a quintuplet, shown in Table II.

Then, the intent goal whose object is the global virtual slice needs to perform further refinement and decomposition until the intent goal's attribute is “operational goal” as shown in Table IV. This process is realized by Pyke with the help of knowledge. It will be passed to the algorithmic model to narrow down the decision space for it, and eventually the DQN will find out the specific management operation parameters.

Based on the intent goals decomposition and translation as described above, the near-RT RIC will create initializing slices on open RAN. During the whole process, the xApp-Monitor in the controller monitors the status of related elements, and stores them in Knowledge. The trained model with the DQN supports the xApp-Slice to realize the resource management through this status. And the performance and gain of our framework will be illustrated by the following two parts:

DQN training performance: The simulation parameters of DQN are shown in Table III. And the Fig. 7 illustrates the convergence of resource management DQN algorithms. In Section V, during the training process of the DQN, we have specified a loss function, also referred to as cost, which decreases gradually. Meanwhile, we investigated the effect of the learning rate for the loss function on our algorithms, and it is shown in the four subplots of Fig. 7. So it is obvious that when the learning rate is too low in Fig. 7(a), the loss function changes slowly and it takes longer to make the algorithm converges. When the learning is too big in Fig. 7(c) and Fig. 7(d), the cost vibration amplitude is very large and may cause the loss function to cross directly over the global optimum point. Therefore, 0.001 is chosen as the suitable learning rate of our algorithm. In Fig. 7(b), during the offline training process to obtain the resource management solution, due to the phenomenon of exploration, the performance of algorithms in the early stages is powerless. The agents tend to rely less on what they have learned and more on taking random actions. At the end of the training, the error is minimised, implying that the Q-value becomes accurate.

The performance gain of delay and rate: The delay variation over time for the open RAN with UE1 as the end-device is shown in Fig. 8(a). And the variation of download rate is shown in Fig. 8(b). It is apparent that at $t = 0$, we enter the intent for automatic driving. Since no automatic driving service slice was previously running on managed entities, the execution starts the slice deployment through the near-RT RIC. After the deployment, UE1 starts to access this service and receives download traffic in the slice.The monitor in the controller shows that the end-to-end (E2E) slice networks have a delay of 37ms and a download rate of 1.8Mbps. These values do not meet the expected results. So the plan first is to find the available rules in knowledge, but no luck. And its DQN module needs to find out the best resource management rules to make the expected results of the system. As a result, the delay and rate began to fluctuate. After a period of training, the system finds the most appropriate rule to distribute by



(a) Delay variation.

(b) Rate variation.

(c) Loss function.

(d) Training reward.

Fig. 8. Performance gain of presented CL control and management.

execution. So the delay and rate of service's slices were stale at the desired value. At this point, the management framework has made the open RAN minimize the delay on the basis of meeting the rate requirements. Through the simulation results of Fig. 8, especially the variation of the loss function in Fig. 8(c), we can also see that the value of ε in the DQN will affect the learning speed of the management system. Either too large or too small values of ε will slow down the management speed. So in our model, we use $\varepsilon = 0.45$ to ensure the system timeliness. Fig. 8(d) shows that the cumulative reward increases as the growth of the number of training episodes, that illustrates the gain of our algorithm.

VII. Conclusion

In this paper, we proposed an intent-driven closed-loop control and management framework and introduced its workflow on the open RAN. To realize closed-loop control, we modeled the problems of the intent goal decomposition and the resource management of slices with event calculus and markov decision process. Addtionally, we designed the deep Q network algorithm to address the problems of resource management. With the help of open source platform, we deployed every control and resource module in our framework. And through the case of "providing automatic driving service", we demonstrated the reliability and performance gain of our framework. It can be used to assist the communication service providers in offering diverse services to consumers and managing them in a highly complex mobile network. In the future, we plan to extend our framework by tailoring the management rules to the corresponding network slices in core network and migrating the management systems to the elements of 6G network. These extensions will further enhance the capabilities of our framework and enable it to adapt to future network requirements.

References

[1] L. Ma, M. Liu, C. Li, Z. Lu and H. Ma, "Cloud-side collaborative computing task scheduling algorithm for 6G edge networks," *J. Beijing Univ. Posts Telecommun.*, vol. 43, no. 6, pp. 66–73, Dec. 2020.

[2] Z. Zhang *et al.*, "6G wireless networks: Vision, requirements, architecture, and key technologies," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 28–41, Sept. 2019.

[3] Y. Gu, Z. Chang, M. Pan, L. Song and Z. Han, "Joint Radio and Computational Resource Allocation in IoT Fog Computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7475-7484, Aug. 2018.

[4] G. Liu, J. Deng, Q. Zheng, G. Li, X. Sun and Y. Huang, "6G smart endogenous: Technical challenges, architecture and key features," *Mob. Commun. 6G Top.*, vol. 45, no. 4, pp. 68–78, 2021.

[5] Y. Xue, L. Yang, and F. Xie, "New user interface design and key technologies in 6G," *Mob. Commun. 6G Top.*, vol. 46, no. 6, pp. 2–7, 2022.

[6] V. Ziegler *et al.*, "6G architecture to connect the worlds," *IEEE Access*, vol. 8, pp. 173508–173520, Sept. 2020.

[7] O-RAN Alliance, Accessed 2022. [Online]. Available: https://www.o-ran.org

[8] 3GPP, "Technical specification group services and system aspects; Service requirements for the 5G system; Stage 1 (Release 17)," *3GPP TS 22.261 V17.11.0*, Sept. 2022.

[9] B. Zong, C. Fan, X. Wang, X. Duan, B. Wang and J. Wang, "6G technologies: Key drivers, core requirements, system architectures, and enabling technologies," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 18–27, 2019.

[10] Open Data Center Committee, "Intention network technology and application white paper," Accessed 2019. [Online]. Available: http://www.opendatacenter.cn/download/24/

[11] P. Zhang, W. Xu, H. Gao and K. Niu, "Toward wisdom-evolutionary and primitive-concise 6G: A new paradigm of semantic communication networks," *Engineering*, vol. 8, no. 6, pp. 60–73, Jan. 2022.

[12] D. Lenrow, "Intent: Don't tell me what to do!" Accessed 2015. [Online]. Available: https://www.sdxcentral.com/articles/contributed/network-intent-summit-perspective-david-lenrow/2015/02/

[13] Open Networking Foundation TR-521, "Intent NBI: Definition and principles," Accessed 2016. [Online]. Available: https://www.opennetworking.org/software-defined-standards/archives/

[14] Gartner, "Innovation insight: Intent-based networking systems," Accessed 2017. [Online]. Available: https://www.gartner.com/doc/3599617/innovation-insight-intentbased-networking-systems

[15] Y. Qie, Y. Li, Y. Ding and C. Zhang, "Intentional awareness network helps the development of the Internet of Things," *Internet Things Technol.*, vol. 12, no. 6, pp. 132–136, 2022.

[16] I. Vaishnavi and L. Ciavaglia, "Challenges towards automation of live telco network management: Closed control loops," in *Proc. 16th Int. Conf. Netw. Serv. Manag. (CNSM)*, Izmir, Turkey, Nov. 2020, pp. 1–5.

[17] ETSI, "Zero-touch network and service management (ZSM); Closed-loop automation; Enablers," *ETSI GS ZSM 009-1 v1.1.1*, 2021.

[18] P. Zhang, K. Niu, and H. Tian, "Technology prospect of 6G mobile communications," *J. Commun.*, vol. 40, no. 1, pp. 141–148, 2019.

[19] 3GPP, "Technical specification group services and system aspects; Telecommunication management; Study on scenarios for intent driven management services for mobile networks (Release 17)," *3GPP TR 28.812 V17.1.0*, Dec. 2020.

[20] Internet Research Task Force, "Draft-irtf-nmrg-ibn-concepts-definitions," Accessed 2018. [Online]. Available: https://irtf.org/

[21] F. Aklamanu, S. Randriamasy, E. Renault, I. Latif and A. Hebbar, "Intent-based real-time 5G cloud service provisioning," in *Proc. IEEE GLOBECOM Workshops (GC Wkshps)*, Abu Dhabi, United Arab Emirates, Dec. 2018, pp. 1–6.

[22] K. Abbas, M. Afaq, T. Ahmed Khan, A. Rafiq and W. -C. Song, "Slicing the core network and radio access network domains through intent-based networking for 5G networks," *Electronics*, vol. 9, no. 10, pp. 1710, Oct. 2020.

[23] W. Chao and S. Horiuchi, "Intent-based cloud service management," in *Proc. 21st Conf. Innov. Clouds Internet Netw. Wkshps. (ICIN)*, Paris, France, Feb. 2018, pp. 1–5.

[24] E. Chirivella-Perez, P. Salva-Garcia, R. Ricart-Sanchez, J. A. Calero and Q. Wang, "Intent-based E2E network slice management for industry 4.0," in *Proc. Jt. Eur. Conf. Netw. Commun. & 6G Summit (EuCNC/6G Summit)*, Porto, Portugal, Jun. 2021, pp. 353–358.

[25] Xinxin Liu, Han Zhao, Miao Pan, Hao Yue, Xiaolin Li and Y. Fang, "Traffic-aware multiple mix zone placement for protecting location privacy," in *Proc. IEEE INFOCOM.*, Orlando, FL, USA, 2012, pp. 972-980.

[26] T. A. Khan, A. Mehmood, J. J. Diaz Ravera, A. Muhammad, K. Abbas and W. -C. Song, "Intent-based orchestration of network slices and resource assurance using machine learning," in *Proc. IEEE/IFIP Netw. Oper. Manag. Symp.*, Budapest, Hungary, Apr. 2020, pp. 1–2.

[27] P. Henrique Gomes *et al.*, "Intent-driven closed loops for autonomous networks," *J. ICT Stand.*, vol. 9, no. 2, pp. 257–290, 2021.

[28] M. Xie *et al.*, "An exposed closed-loop model for customer-driven service assurance automation," in *Proc. Jt. Eur. Conf. Netw. Commun. & 6G Summit (EuCNC/6G Summit)*, Porto, Portugal, Jun. 2021, pp. 419–424.

[29] N. F. S. d. Sousa and C. E. Rothenberg, "CLARA: Closed loop-based zero-touch network management framework," in *Proc. IEEE Conf. Netw. Funct. Virt. Softw. Defined Netw. (NFV-SDN)*, Heraklion, Greece, Nov. 2021, pp. 110–115.

[30] B. Angui, R. Corbel, V. Q. Rodriguez and E. Stephan, "Towards 6G zero touch networks: The case of automated cloud-RAN deployments," in *Proc. IEEE 19th Ann. Consum. Commun. & Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2022, pp. 1–6.

[31] A. C. Baktir, A. N. Junior, and A.Zahemszky"Intent-based cognitive closed-loop management with built-in conflict handling," in *Proc. IEEE 8th Int. Conf. Netw. Soft. (NetSoft)*, Milan, Italy, 2022, pp. 73–78.

[32] T. Li, L. Zhao, F. Song and C. Pan, "OAI-based end-to-end network slicing," in *Proc. IEEE 23rd Int. Conf. Digit. Signal Process. (DSP)*, Shanghai, China, Nov. 2018, pp. 1–4.

[33] E. Ramya and R. Gobinath, "Reducing end to end delay through adaptive greedy search algorithm for wireless body area sensor network," *Int. J. Recent Technol. Eng. (IJRTE)*, vol. 8, no. 2S4, pp. 2277–3878, Jul. 2019.

[34] J. Gao *et al.*, "Resource allocation for network slices with delay-sensitive multimedia services," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Paris, France, Oct. 2020, pp. 1–5.

[35] M. Roddy *et al.*, "5G network slicing for mission-critical use cases," in *Proc. IEEE 2nd 5G World Forum (5GWF)*, Dresden, Germany, Sept./Oct. 2019, pp. 409–414.

[36] S. R. White, J. E. Hanson, I. Whalley, D. M. Chess and J. O. Kephart, "An architectural approach to autonomic computing," in *Proc. Int. Conf. Auton. Comput. (ICAC '04)*, New York, NY, USA, May 2004, pp. 2–9.

[37] 3GPP, "Technical specification group radio access network; Study on new radio access technology; Radio access architecture and interfaces (Release 14)," *3GPP TR 38.801 v14.0.0*, Mar. 2017.

[38] R. Schmidt, M. Irazabal and N. Nikaein, "FlexRIC: An SDK for next-generation SD-RANs," in *Proc. 17th Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT '21)*, Virtual Event, Munich, Germany, Dec. 2021, pp. 411–425.

[39] J. Zhang, C. Yang, A. Anpalagan, R. Dong and L. Zhang, "IDSM: Intent-Driven Slice Management and Maintenance for 6G RANs," in *Proc. IEEE Int. Conf. Commun. Technol. (ICCT)*, Nanjing, China, 2022, pp. 1244–1249.

[40] D. Ye, R. Yu, M. Pan and Z. Han, "Federated Learning in Vehicular Edge Computing: A Selective Model Aggregation Approach," *IEEE Access.*, vol. 8, pp. 23920–23935, 2020.

[41] L. Li, D. Shi, R. Hou, H. Li, M. Pan and Z. Han, "To Talk or to Work: Flexible Communication Compression for Energy Efficient Federated Learning over Heterogeneous Mobile Edge Devices," in *Proc. IEEE INFOCOM 2021. IEEE Conf. Comput. Commun.*, Vancouver, BC, Canada, 2021, pp. 1–10.

Jingwen Zhang received the bachelor's degree in communication engineering from Ningbo University, Ningbo, China, in 2021, where she is currently pursuing the master's degree in information and communication engineering. She is also with GUIDE Research Team, which is guided by Dr. C. Yang. Her research interests include mobile communication networks and reinforcement learning.

Chungang Yang (Senior Member, IEEE) received the bachelor's and Ph.D. degrees from Xidian University, Xi'an, China, in 2006 and 2011, respectively. From September 2010 to March 2011, he held the Visiting Scholar position with the Department of Electrical and Computer Engineering, Michigan Technological University (MTU). From March 2015 to March 2016, he held the Visiting Scholar position with the Department of Electrical and Computer Engineering, University of Houston (UH). He is currently a Full Professor with Xidian University, where he leads the research team of GUIDE (Game, Utility, artificial Intelligent Design for Emerging communications). His recent research interests include artificial intelligent B5G/6G wireless mobile networks, intent-driven/based networks (IDN/IBN), and space-terrestrial networks (STN).

Ru Dong received the bachelor's degree in communication engineering from Xi'an University of Architecture and Technology, Xi'an, China, in 2020, where she is currently pursuing the master's degree in information and communication engineering. She is also with GUIDE Research Team, which is guided by Dr. C. Yang. Her research interests include mobile communication networks and 5G core network orchestration.

Yao Wang received the B.E. degree in Internet of Things engineering at the School of Information and Engineering from Baoding University, Baoding, China, in 2019, and the M.S. degree in computer science and technology at the School of Information and Electrical Engineering from Hebei University of Engineering, Handan, China, in 2022. He is currently pursuing the Ph.D. degree in information and communication engineering at the School of Telecommunications Engineering, Xidian University, Xi'an, China. He is also with GUIDE Research Team, which is guided by Dr. C. Yang. His current research interests include 6G, intent-driven networks, goal decomposition, and game theory.

Alagan Anpalagan (Senior Member, IEEE) received the B.A.Sc., M.A.Sc., and Ph.D. degrees in electrical engineering from the University of Toronto, Canada. He has held administrative positions in the department as the Associate Chair, the Program Director of Electrical Engineering. He has received several awards, including the IEEE Canada J. M. Ham Outstanding Engineering Educator Award (2018), SGS Outstanding Contribution to Graduate Education Award (2017), and Faculty Scholastic, Research and Creativity Awards from Ryerson University. He was also the recipient of the IEEE M. B. Broughton Central Canada Service Award (2016), and was a co-author of a paper that received the IEEE SPS Young Author Best Paper Award (2015). He is a Registered Professional Engineer in the Province of Ontario, Canada, a Fellow of the Institution of Engineering and Technology (FIET), and a Fellow of the Engineering Institute of Canada (FEIC).

Qiang Ni (M'04–SM'08) received his Ph.D. degree in engineering from Huazhong University of Science and Technology, Wuhan, in 1999. He is currently a Full Professor and Head of the Communication Systems Research Group at the School of Computing and Communications, Lancaster University, Lancaster, U.K. He has authored over 180 research papers in international journals and conferences, and was an IEEE 802.11 Wireless Standard Working Group Voting Member and a Contributor to the IEEE Wireless Standards. His research interests include future generation communications and networking systems.

Mohsen Guizani (s'8s-m'89-sm'99-f'09) received his M.S. and the Ph.D. in Electrical and Computer engi-neering from Syracuse University, Syracuse, NY, USA. He is currently a Professor and the Associate Provost at Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, UAE. His research interests include applied machine learning, smart city, wireless communications/networking, cloud computing, security, and its application to healthcare systems. He was elevated to the IEEE Fellow in 2009. He is currently serving on the editorial boards of many IEEE transactions and magazines.