A Brief Survey and Implementation on AI for Intent-Driven Network

Jiaorui Huang, Chungang Yang, Shiwen Kou, Yanbo Song
The State Key Laboratory on Integrated Services Networks
Xidian University
Xi'an, China
hjr6286@163.com, chgyang2010@163.com, shiwenkou@gmail.com, songyanbo94@163.com

***Abstract*—Intent-driven network (IDN, or intent-based network, IBN) is a novel networking paradigm, which can enable user intents to drive network management autonomously and improve the network's operational efficiency. Although artificial intelligence (AI) has been found for several applications to the IDN, there lacks a systematic discussion and research on this topic. In this work, we present a survey of the application of AI at each layer of IDN. Then, a general IDN management architecture, State-Action-Intent (SAI), is proposed. The presented SAI is a new IDN implement framework to automate the operational intents in a closed loop to overcome the challenges of complex network services. To verify the availability and effectiveness of SAI, a proof-of-concept demonstration is provided, and the obtained performance is discussed.**

***Index Terms*—Artificial Intelligence, Intent-driven Network, Network Architecture**

I. Introduction

Future network management will be under tremendous pressure with the expanding network scale as the user demands changes with diverse business applications. While the next generation of network updates, network management could be enhanced by two technical routes, which are using smarter and more efficient network architectures and adopting artificial intelligence (AI) - based management algorithms. Intent-driven network (IDN) is an emerging network architecture, which originates from a software-defined network and inherits the characteristics of decoupling the data plane and the control plane and the advantages of facilitating centralized control and management of different types of networks [1]. This architecture ensures that the IDN is flexible, reconfigurable, open, and customizable. Great results were achieved in the intent refinement system [2]. It provides a facilitative interface for interaction between users and the network and plays a very important role in IDN.

As it is known, AI is a new technology science which has been proposed to reduce human interference and hence improve network automation and security for future communication networks. Especially in large-scale network applications become very popular. A lot of work has been done to realize the autonomous management of the network in general [3], [4]. In 2017, the ETSI has proposed a Zero-touch network and Service Management (ZSM) framework to achieve zero-touch management of end-to-end services across different management domains, providing an important tool for closed-loop networks. In [3], closed-loop automation within the ZSM is reviewed briefly and a methodology that uses intents to coordinate hierarchies of closed loops is presented; however, this work lacks a comprehensive description on the application of AI in various layers. Intelligent intent based networks concept and trial results are presented in [4] with the feasibility of automatically assembling self-learning closed loop micro-services into intent specific automation pipelines, however the paper only applies AI specifically to one layer. The purpose of this paper is to fill this gap by providing an overview of the existing the application of AI in each layers of an IDN network in literature.

Hence a comprehensive approach should be considered to combine the advantages of IDN and AI to achieve more efficient network structures and algorithms. The authors in [5] discussed the vision of the future autonomous network and the challenges of improving network performance. A self-management framework and current technologies and remaining challenges are summarized in [6]. The above articles are to provide the autonomous network architecture, but do not give an explicit example of an AI application. In this paper, we provide a novel intelligent autonomous network architecture State-Action-Intent (SAI), in which demands can be accomplished using AI techniques in each layer of the structure. Meanwhile, we apply a use case to verify the availability and effectiveness of SAI.

The main contributions of this article are as follows:

- The purpose of this paper presents a brief survey on AI applications to IDN. Each layer of IDN is divided into three functional modules and described in detail.
- A SAI architecture is proposed, which can be flexibly configured in different scenarios. The architecture is divided into three layers and two operational mechanisms, forming a dual-driven, closed-loop service architecture.
- We provide a use case to verify the availability and effectiveness of SAI.

The rest of this article is organized as follows. In section II, we present a basic background about IDN, the main network architecture and a general view. In section III, we review

This work was supported in part by the National Science Foundation of China (61871454).

relevant work on AI for IDN and present our contributions to this survey. In section IV, we introduce a new intelligent dual-drive network architecture. In section V, we discuss a use case to verify the availability and effectiveness of our proposed architecture. Finally, the conclusions and future work are described in Section VI.

II. A General View on IDN Architecture and Key Technologies

In this section, we provide an essential architecture and a general view on IDN. The IDN architecture consists of three layers and two interfaces. This is well illustrated and demonstrated in Fig. 1.

A. IDN Architecture

Each layer is described in detail as follows.

Business application layer is mainly concerned with collecting user intent and translating it into a uniform format for machine parsing. Among them, user intent contains intrinsic intent generated from within the system, and extrinsic intent entered from outside by the manager or server. Intrinsic and extrinsic intent are expressed in different forms, where the former usually uses network information parameters to reflect the current network situation, and the latter is expressed in more diverse forms, such as voice, text, gestures, etc.

Intent-enabled layer is the core of the IDN. It consists of control, management, and decision-making on the network not only to analyze the results of intent translation but also to enable unified orchestration of the underlying network resources.

Infrastructure layer is a collection of network data capture tools. Data capture tools on the infrastructure layer can collect situational information and policy configuration parameters and can assist intelligent decisions, considering various efficiency metrics, i.e., bandwidth, delay, topology, etc.

The specific definitions of the two interfaces are as follows.

Intent northbound interface (NBI) is one of the key features, for users to access, develop and manage the network. By making use of the controller development of upper layer business, user interaction with the network becomes more friendly. With the diversity of business applications, there is need for diversity, rationality, and openness in NBI.

Intent southbound interface (SBI) is one of key enablers of IDN implementation where decoupling the data plane from the control plane can be enhanced to function as programmable control plane. SBI has a standard interfaces to communicate with the infrastructure layer.

B. A General View on IDN

In the subsections, we share a general view from two aspects. On the one hand, the whole network can be analyzed horizontally into three layers and two interfaces. The function of each component has been discussed in detail in the previous subsection. On the other hand, the whole network architecture can be divided into intent realization and intent guarantee from the vertical analysis, which are analyzed in detail as follows.

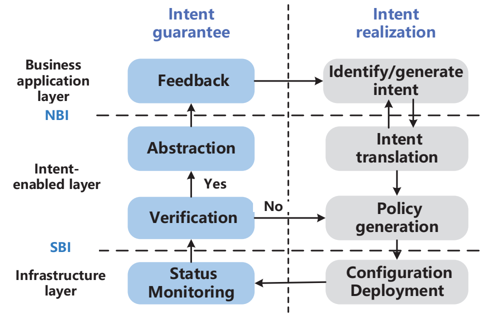


Fig. 1. IDN architecture and key technologies.

The intent realization consists of intent generation, intent translation, policy generation, and configuration deployment. Among them, intent is mainly generated by users, who may be operators, managers or intelligent machines. Later, the intent is converted into a machine-understandable language by a unified specification approach; the system develops a policy that can be executed based on the physical link information and the user intent. The intent guarantee provides guarantees for the intent realization process, which mainly consists of status monitoring, policy verification, abstraction, and feedback. The process obtains network status information through the underlying physical network components to verify whether the network service meets the user's needs and the system feeds the results back to the user, forming a closed loop.

In summary, IDN is a programmable and scalable architecture with its significantly less complexity for network deployment. Its closed-loop design can better respond to the dynamic network state and various user requirements. However, most solutions are based on manual implementation. At present, the application of AI in various fields is blooming. If AI is applied to IDN, it will make IDN more intelligent, concise and efficient. In Section III, a detailed survey on the application of AI for IDN is given.

III. A Brief Survey on AI for IDN

In this section, we conduct an extensive literature review for each category and discuss available techniques from the application of AI in the business application layer, the intent-enabled layer, and the infrastructure layer.

A. Application of AI in Business Application Layer

We provide a brief survey of the business application of AI in IDN, where intent perception, intent translation, and intent conflict detection.

Intent perception refers to extract the potential purpose of users and decompose their abstract intents into easy-to-execute ones. The intent perception data comes from both intrinsic and extrinsic intent. Intrinsic intents are the raw data obtained from the internal infrastructure layer of the network, and extrinsic intents are representing in the form of text and voice. Recently, two approaches were mentioned in [7]: Natural Language Processing (NLP)-based language analysis and rule-based parsing algorithm. The former can allow various forms of input, but it is difficult to implement; the latter is set according to pre-set rules, which is easy to implement but only has a single input form.

Intent translation means to resolve users' intents into the corresponding network configuration policies. The translation can be classified into template-based and syntax-semantic-based. Among them, template-based translation requires users input depending on the template, while the translation based on syntax and semantic aims to extract semantic information through the syntax of different types of languages. In [8], an intent engine, iNDIRA, is described to provide reliable and simple communication between users and the network. It mainly uses natural language and ontology to convert definition queries into corresponding network commands, and then reconfigure resources and policies. Therefore, the users' intents can be better presented through iNDIRA.

Intent conflict detection is designed to prevent disagreements between multiple intents, resulting in network services that do not fulfill the reality of the users' needs. Intent conflicts can be divided into two categories: direct and indirect. Direct conflicts are easy to detect, while indirect conflicts are more difficult to detect. In [9], a supply chain network scenario is considered where an intent based networking is employed to address the complexity, dynamics and heterogeneity of the network. The proposed controlled natural languages technology is used to resolve the conflict between intents and prevent the generation of invalid policies.

B. Application of AI in Intent-enabled Layer

The intent-enabled layer is mainly responsible for controlling, managing and making decisions in the network. The application of AI in the intent-enabled layer is mainly discussed and analyzed from the following three aspects: policy refinement, policy mapping, and policy conflict detection.

Policy refinement focuses on splitting high-level abstract policies into executable configurations. Methods of refinement can be segmented into three types: rule-based transformation, classification-based refinement and case-based reasoning, and logic-based approaches. Of which, the first one is refined according to pre-set rules. The second is mainly implemented with the help of a range of classification techniques. The third is based mainly on the research of reasoning to accomplish the division of policies, which has less automation. Currently, Karimi et al. [10] propose an attribute based access control policy extraction method based on unsupervised learning. This method can not only filter all aspects of the policy, improve the refinement accuracy, but also complete the work in the case of incomplete access log records.

Policy mapping is the process of matching the intent to the best policy in the policy repository. According to the scope of mapping, it can be split into local and global. Local mapping refers to constructing cost functions within the range of optional strategies, and global mapping is to build energy functions using global optimization theory. In [11], an intent defined optical network architecture is proposed. The architecture focuses on a multi-feature extraction method based on a deep neural network algorithm, which can quickly extract diverse features of different services and accurately identify service intent. It uses Long short-term memory (LSTM) to extract keywords and map user intent to the corresponding network constraint parameters. After that, it connects to network components through the Openflow protocol to model the mapping relationship between services and intents.

Policy conflict detection is mainly to avoid blocking in the system and to check the policy in advance. Based on the intersection of policy matching domains and the actions performed by the policies, several conflict relations of policies are given: mutual exclusion, redundancy and override. Yang et al. [11] argue to focus on a guarantee mechanism based on the deep neural evolutionary network (DNEN), which can extract depth fault features and accurately find the real failures when a large number of alarm information appears. DNEN can continuously change and modify the weights during the training process. Therefore, it can go beyond the local optimum to find the global one, thus it can better locate failures during the network intent configuration and ensure highly accurate fault location in large-scale networks.

C. Application of AI in Infrastructure Layer

This subsection describes the application of AI in the infrastructure layer is mainly discussed in three aspects: policy deployment, policy verification, and situation awareness.

Policy deployment refers to running the generated policy in the current network environment. The operation of the policy drives the allocation and scheduling of resources in the network. Here, resources include three aspects, the first is physical resources; the second is the platform; the third is the service function chain. In [12], a new system, LUMI, enables operators to talk with the network. The input of the system is a natural language expression of intent, which is encoded and labeled by bi-directional LSTM and conditional random field entities, enabling intent-to-instruction conversion. And then, the system maps the commands into Merlin to manage and control the network resources. Therefore, the system realizes the whole process from intent input to resource deployment.

Policy verification means verifying that the deployed policy not only meets the needs of users but also falls within the bearer range of the network. It is analyzed and verified in three aspects: the rationality of resource allocation, the feasibility of policy deployment, and the correctness of policy execution. Bonfim et al. [13] proposed an automatic policy refinement system. The system contains a semantic verification module that analyzes policies by using a description logic semantic reasoner. Moreover, the verification of policy enforceability is achieved by using a description logic inconsistency validation method. Ultimately, the whole system allows for formal verification of network services.

Situation awareness is meant to predict future trends and thus collect information on network dynamics. It can be divided into active and passive. The former collects information when the network is operating normally; the latter collects information when the network is abnormal to infer the cause

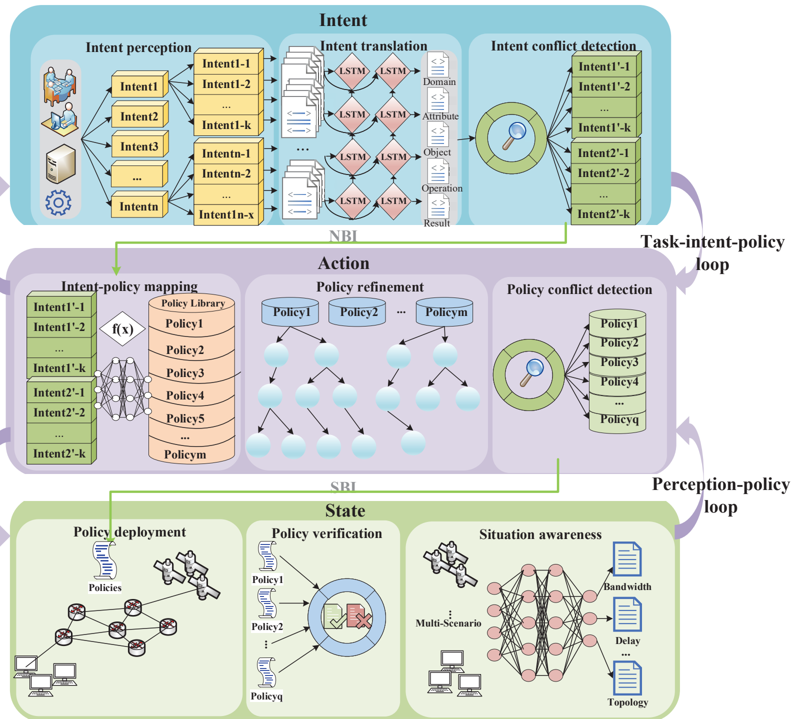


Fig. 2. SAI architecture.

of the failure. Khan et al. [14] introduce a neural network multilayer perceptron (NNMLP) in the network architecture, which is used to periodically receive the resource state and update the system according to future state requirements. Abbas et al. [15] employ an intelligent update and assurance engine to handle scalability and assurance of runtime resources. This mechanism uses a recurrent neural network LSTM model to predict resource utilization and update the overall system configuration parameters.

IV. A SAI Based on IDN Refinement

In this section, we propose a new architecture State-Action-Intent (SAI). The architecture is simple and straightforward, as shown in Fig. 2.

A. The Architecture of SAI

SAI is an emerging concept in networks to automate a number of operational tasks in closed-loop to overcome the challenges of complex network services. We describe each part in detail as follows.

Intent is derived from user needs and ideas to drive the network to perform a series of operational services to overcome the changes of user demands in complex scenarios. To provide better services for users, the network needs to constantly mine and analyze users' intents. We divide the process of intent into intent perception, intent translation, and intent conflict detection. Among them, intent perception can use Generative Adversarial Networks (GAN) to perform data mining. Intent translation is usually implemented by using, for example, NLP, GAN, LSTM, etc. Intent conflict detection can be selected from multilayer perceptron for analysis and processing.

Action refers to the process of developing a policy for the realization of user intent. Facing different problems in various scenarios, the generation of policies is similar to the process from zero to one. But in most cases, the completion of intent requires a combination of multiple policies to form a new policy, which reflects the idea of from zero to one. We divide the policy generation process into policy mapping, policy refinement, and policy conflict detection. In policy mapping, the existing neural network and LSTM can implement. In policy refinement, unsupervised learning can be handled well. In policy conflict detection, DNEN can accurately find the real fault according to the fault characteristics.

State refers to the perception, understanding, and prediction of the operating state of the network. Typically situation awareness is performed by the network sensors deployed in the network for information acquisition. With the help of state information, administrator could better manage the network. Situation prediction is to plan and deploy the network status in advance to better maintain the normal operation of the network. Therefore, we divide the situation awareness into policy deployment, policy verification, and situation awareness. Through research, it is found that AI has many methods in situation awareness. For example, GAN, NNMLP, LSTM, etc. can all be used for network prediction.

B. Working Mechanisms

In this subsection, top-down task-intent-policy loop and bottom-up perception-policy loop are the two core working

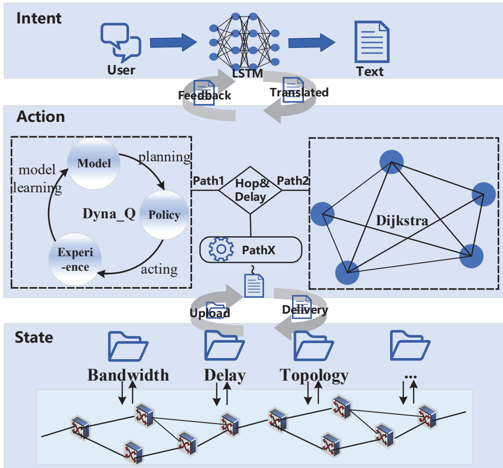


Fig. 3. An auxiliary path generation use case architecture.

mechanisms are discussed.

Task-intent-policy loop is one of the cores of the SAI, and it is also the key operating mechanism for the entire architecture to achieve closed-loop self-driving. The loop runs through the intent translation on the Intent and sends the results down to the Action via NBI. Action maps the received translation results to the policies in the policy library to generate network policies that can be safely executed. Generated policies are delivered to State via SBI for policy deployment and enforcement. The loop can greatly improve the sense of user-system interaction while reducing human influence in the circuit and facilitating the automation of the network.

Perception-policy loop is one of the important key technologies of SAI, which provides a guarantee for the closed-loop automation of the architecture. The loop runs from the bottom up, and the State obtains network situation information and sends it back to Action through SBI. Action makes comprehensive analysis and judgments based on current network conditions and user intent, and then makes dynamic adjustments to network policies. The system feeds back the results of policy adjustment to users and network physical devices through NBI and SBI, respectively, to form a closed-loop. The loop is driven by the situation and can facilitate autonomous decision-making and adjustment of the network, which is a key mechanism for network closure automation.

V. Proof-of-concept Demonstration of SAI

In this section, we discuss a use case to verify the availability and effectiveness of SAI. This use case is to enable intelligent auxiliary path generation to assist administrators in decision making.

A. Use Case Architecture

The use case architecture is shown in Fig. 3. In Intent, users generate intents on demand and translates the natural language into a language that can be understood by machines.

This part can refer to the previous research [2] of our research group. Therefore, in this use case, it is considered that the intent has been accurately translated, and then the translation result is sent to the Action to drive the formulation and generation of the policy. At the same time, the State stores the perceived bandwidth, delay, and topology information in real time, and uploads it to Action. In Action, different path policies are formulated based on user intent and real-time perception information.

After the path policy is issued to State for execution, the result of the execution is fed back to the user, forming two loops. The intent-driven and situation-driven mechanisms are well demonstrated.

B. Algorithm Design

In the subsection, the algorithms for each part of the use case architecture are designed in detail as follows.

State mainly senses the topology, bandwidth and delay of the network. Network topology is obtained by the controller sending link layer discovery protocol (LLDP) packets. Network bandwidth is obtained by using the OpenFlow protocol to obtain port statistics. The delay between switch nodes is calculated by using the transceiver timestamp of LLDP messages and the controller measured by echo messages to switches.

Action mainly use Dyna-Q algorithm and Dijkstra algorithm to realize auxiliary path formulation. Dyna-Q algorithm combines model-based and non-model-based intensive learning. Dijkstra is a typical shortest path routing algorithm that computes the shortest path from one node to all other nodes, which is not described in detail here. The following is a detailed description of the Dyna-Q algorithm.

In the Dyna-Q, bandwidth and delay are considered to calculate the reward function. The specific expression of the reward function is

$$Reward = \alpha Delay + \beta Bandwidth, \tag{1}$$

where α and β are weights. The $Delay$ is specifically expressed

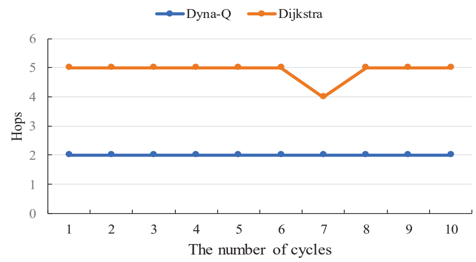
$$Delay = \frac{2}{\pi} arctan(d_{mn} - \frac{\sum\limits_{d_{mi} \in TP_m} d_{mi}}{|TP_m|}), \tag{2}$$

where d_{mn} represents the element of m row and n column of the delay. TP_m represents the total number of links in m row of the topology. $\sum\limits_{d_{mi} \in TP_m} d_{mi}$ represents the sum of the delays corresponding to the connected links in m row. The $Bandwidth$ is specifically expressed

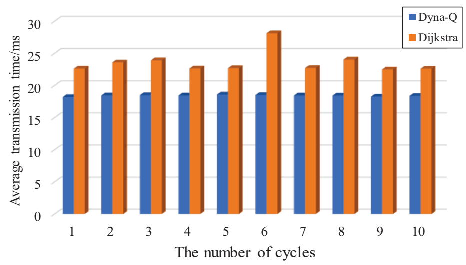
$$Bandwidth = \frac{2}{\pi} arctan(0.01(b_{mn} - \frac{\sum\limits_{b_{mi} \in TP_m} b_{mi}}{|TP_m|})), \tag{3}$$

where b_{mn} represents the element of m row and n column of the bandwidth. $\sum\limits_{b_{mi} \in TP_m} b_{mi}$ represents the sum of the bandwidths corresponding to the connected links in m row.

Intent mainly refers to the intent generated by the user. We consider that that the intent has been accurately translated, and the translation result is sent to the Action to drive the formulation and generation of the policy.



(a) The average duration of intent refinement.



(b) Link packet loss rate over time.

Fig. 4. The performance of the presented SAI.

C. Realization

We use python 2.7 to implement the full process of SAI. The network topology is defined in Mininet. Mininet is a virtualized network simulation tool that can create a virtual network containing hosts, switches, controllers, and links. As for the SDN controller, we used RYU, which controls switches through the OpenFlow protocol. We simulated a communication network with 14 network nodes, and each network node corresponds to a switch, and each switch has a host connected. Therefore, path selection in the network can be considered as a routing problem between switches.

D. Result Analysis

The use case senses information about the network data in real time to make the appropriate network path selection. We use the RYU controller, which is based on logical sequential cycle execution. We tested the network dynamic real-time parameters between two fixed nodes in 10 cycles and generated the corresponding path policies during each cycle. The Dyna-Q algorithm and Dijkstra's algorithm are used to calculate the paths. In view of the Fig. 4, a comparative analysis of the two performance aspects, the number of path hops and the average transmission time, is provided. From Fig. 4(a) we can get the number of path hops calculated by the Dyna-Q algorithm is significantly less than it calculated by the classical Dijkstra algorithm. Meanwhile, the Ping tool was used to test the transmission time between two nodes, and it is clear that the average transmission time of Dyna-Q is lower than the average transmission time of Dijkstra's algorithm, as shown in Fig. 4(b). In the experiments, we also conducted tests between two random nodes, and the overall results of the tests showed that the algorithm supported by AI performs relatively well. Therefore, we introduce AI into the network architecture to optimize configuration and efficient use of resources, while allowing the network to have more available resources to continuously upgrade and develop new services.

VI. Conclusions

We presented an systematic survey on the application of AI in each layers of IDN. We first summarized the motivation for the application of AI in networks, which leads to a brief survey on AI applications to the different modules of the various layers of the IDN, and we proposed a new network architecture SAI which will enable the application and implementation of AI at all layers of the IDN. To verify the availability and effectiveness of SAI, a proof-of-concept demonstration was proposed, and the obtained performance was discussed.

References

[1] L. Pang, C. Yang, D. Chen, Y. Song, and M. Guizani, "A Survey on Intent-Driven Networks," *IEEE Access*, vol. 8, pp. 22862–22873, 2020.

[2] Y. Ouyang, C. Yang, Y. Song, X. Mi, and M. Guizani, "A Brief Survey and Implementation on Refinement for Intent-Driven Networking," *IEEE Network*, vol. 35, no. 6, pp. 75–83, 2021.

[3] P. H. Gomes, M. Buhrgard, J. Harmatos, S. K. Mohalik, D. Roeland, and J. Niemöller, "Intent-driven closed loops for autonomous networks," *Journal of ICT Standardization*, pp. 257–290, 2021.

[4] P. Szilágyi, "I2bn: Intelligent intent based networks," *Journal of ICT Standardization*, pp. 159–200, 2021.

[5] P. Harvey, A. Tatar, P. Imai, L. Wong, and L. Bringuier, "Evolutionary Autonomous Networks," *Journal of ICT Standardization*, pp. 201–228, 2021.

[6] M. Bezahaf, S. Cassidy, D. Hutchison, D. King, N. Race, and C. Rotsos, "A Model-Driven and Business Approach to Autonomic Network Management," *Journal of ICT Standardization*, pp. 229–256, 2021.

[7] A. Banerjee, S. S. Mwanje, and G. Carle, "An intent-driven orchestration of cognitive autonomous networks for ran management," in *2021 17th International Conference on Network and Service Management (CNSM)*, pp. 380–384, IEEE, 2021.

[8] M. Kiran, E. Pouyoul, A. Mercian, B. Tierney, C. Guok, and I. Monga, "Enabling intent to configure scientific networks for high performance demands," *Future Generation Computer Systems*, vol. 79, pp. 205–214, 2018.

[9] M. Bensalem, J. Dizdarević, F. Carpio, and A. Jukan, "The Role of Intent-Based Networking in ICT Supply Chains," in *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, pp. 1–6, IEEE, 2021.

[10] L. Karimi, M. Aldairi, J. Joshi, and M. Abdelhakim, "An automatic attribute based access control policy extraction from access logs," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[11] H. Yang, K. Zhan, Q. Yao, X. Zhao, J. Zhang, and Y. Lee, "Intent defined optical network with artificial intelligence-based automated operation and maintenance," *Science China Information Sciences*, vol. 63, pp. 1–12, 2020.

[12] A. S. Jacobs, R. J. Pfitscher, R. H. Ribeiro, R. A. Ferreira, L. Z. Granville, W. Willinger, and S. G. Rao, "Hey, Lumi! Using Natural Language Intent-Based Network Management," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 625–639, 2021.

[13] M. Bonfim, F. Freitas, and S. Fernandes, "A Solution for Goal-oriented Policy Refinement in NFV Management and Orchestration Systems," in *Anais Estendidos do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pp. 73–80, SBC, 2021.

[14] 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 *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–2, 2020.

[15] K. Abbas, T. A. Khan, M. Afaq, and W.-C. Song, "Network Slice Lifecycle Management for 5G Mobile Networks: An Intent-Based Networking Approach," *IEEE Access*, vol. 9, pp. 80128–80146, 2021.