NetGraph: An Intelligent Operated Digital Twin Platform for Data Center Networks

Hanshu Hong, Qin Wu, Feng Dong, Wei Song, Ronghua Sun, Tao Han
Huawei Technologies

Cheng Zhou, Hongwei Yang
China Mobile

ABSTRACT

This paper presents a digital twin platform towards automatic and intelligent management for data center networks: NetGraph. It builds a virtual image of the physical network, which comprises the entire network elements and simplifies the workflows of network service management. NetGraph redefines network operation (including maintenance) model by integrating four functional blocks: unified digital twin model management, automatic configuration deployment and translation, Inventory search, and network elements validation. NetGraph has been deployed in Huawei's data center networks and served more than fifty thousand devices along with millions of network connectivity information (e.g., links and terminations points). Statistics in real network scenarios show that NetGraph has a wide coverage of network models and supports multiple functions.

CCS CONCEPTS

• **Networks** → **Network management**.

KEYWORDS

network model, automation, management, deployment

ACM Reference Format:
Hanshu Hong, Qin Wu, Feng Dong, Wei Song, Ronghua Sun, Tao Han and Cheng Zhou, Hongwei Yang. 2021. NetGraph: An Intelligent Operated Digital Twin Platform for Data Center Networks. In *ACM SIGCOMM 2021 Workshop on Network-Application Integration (NAI '21), August 27, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3472727.3472802

1 INTRODUCTION

1.1 Background

With the fast global expansion of networks and the increased demand placed on data center networks, dynamically adapting to customer needs (e.g., zero packet loss, high performance computing, low latency) becomes a big challenge to network operators, e.g., scaling data centers on demand. The data center network may consist of heterogeneous devices from different vendors, which are equipped with different configuration commands. One conventional method to configure these devices is to use vender-specific command-line interfaces (CLIs). However, such a method is not optimal within the context of complex, multi- functional services. Specifically, the equipment types and configuration commands vary from one vendor to another, network managers would have to acquire the appropriate vendor-specific expertise. This inevitably aggravates the burden of network managers and fails to satisfy the need for fast service deployment [3]. With thousands of switches and servers, the ability to conduct advanced network management, thus, minimizing the time and work force consumption becomes a huge challenge. The concept of digital twin was proposed by Grieves et. al [14]. It is defined as a virtual representation that serves as the real-time digital counterpart of a physical entity and reflects the whole life- cycle device management [16]. A digital twin network platform is designed by applying digital twin techniques to communication networks: it creates a virtual image of a physical network. By taking advantage of such platform, the efficiency of the physical network management can be effectively promoted [17]. Nevertheless, there are several challenges raised by the design and the deployment of constructing digital twin network in DCN scenario:

Complexity of modelling. Data modeling is the core component of the digital twin network construction. Considering the large amounts of data that represent a physical network, data modelling has to accurately reflect the various functions supported by the network, but also be flexible and extensible.

Inaccurate and inefficient mapping. It is difficult to map the unified data models used by a digital twin network to the corresponding vendor-specific devices, especially in terms of the instantiated data, such as configuration files of devices and network services.

Effort and skill-intensive work for network information retrieval. The digital twin network platform will have to manipulate large amounts of data as a function of the nature and the scale of the network, let alone the nature of the services it supports. Identifying and retrieving the relevant information for any given network management operation may prove to be difficult [5].

1.2 Motivation and Contribution

To address the aforementioned challenges, we develop a new digital twin network platform, which is also referred to as an intelligent operated digital twin platform. The platform satisfies the following requirements:

Full network modelling: current modelling works mainly focus on the entire network using configuration files [7]; this method separates the configuration data from state data. As a consequence, it can hardly conduct correlation analysis during daily network management. A digital twin model that comprises both configuration and network state data is therefore required.

NAI '21, August 27, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8633-3/21/08…$15.00
https://doi.org/10.1145/3472727.3472802

Intelligent management and deployment: many network changes [4] require the manager to create a new change request and respond to it manually. The more change requests over a given period of time, the heavier the workload. Network management processes should therefore be simplified, e.g., by means of intent- based networking where the network operator expresses an intent that will be dynamically interpreted by the digital twin platform and then, appropriate configuration tasks will be automatically executed on the involved network devices.
Efficient data retrieval: the digital twin platform shows a virtual image of the whole network and includes huge amounts of data. Efficient data retrieval with inventory search can help users quickly acquire the network state information.

This paper illustrates the advantage of digital twin techniques and presents NetGraph, a digital twin network platform that facilitates intelligent network maintenance. Our design introduces three digital twin models to describe the network state and simplify network management: device, network, and service models.

Whenever a network change occurs, managers only need to modify the digital twin models and subsequent configuration tasks are automatically conducted by NetGraph. In addition, NetGraph provides multiple functional blocks such as inventory search and models translation. These functional blocks support efficient network information retrieval, services migration and effective network element validation. NetGraph has been deployed in Huawei's DCN, and the real network statistics demonstrate its high performance.

The rest of the paper is organized as follows: Section 2 introduces the typical application scenarios and functional blocks of NetGraph. Section 3 provides an overview of the NetGraph architecture. Section 4 introduces the methods adopted to deploy NetGraph in a real network. Section 5 provides some evaluation results of NetGraph in terms of performance and functionality. Section 6 discusses the current limitations and future research directions. Section 7 reviews the relevant literature. Section 8 concludes the paper.

2 NETGRAPH: APPLICATION SCENARIOS

NetGraph includes four key functional blocks to support real use cases in Huawei's data center networks:
Unified model management: a complex network is abstracted into three types of digital twin models: device, network, and service models. The device model abstracts the state of a physical device by leveraging configuration files and other state-related data. The network model illustrates the topology of the network including the connection relationships between devices, ports, and protocols at different layers. The service model describes a service supported by the network, and maps the service to both the device and network models. NetGraph models a DCN in a vendor-agnostic fashion and provides the lifecycle management of these models. First, the device network model is constructed from data stored in configuration files and also from data that reflect the network state. Whenever these data change, NetGraph will also update the models automatically. Second, NetGraph records all the operations applied to these models by creating a log file with time stamping. When data change, NetGraph refreshes the log entry and its timestamp to ensure the full cycle monitoring of these models. Third, NetGraph supports secure rollbacks of these models whenever a network failure occurs.
Automatic configuration deployment and translation: NetGraph supports a bidirectional mapping between models and configuration files. On one hand, device and network models can be derived from configuration files as well as state data from the status of the real network [12]; On the other hand, NetGraph can translate the device and network models into corresponding configuration files by using the model values and vendor specific templates. NetGraph parses the configuration files that are retrieved from the real network devices. It also abstracts the status of the network and then analyzes the configuration files by comparing the network status with the target status in terms of compliance, consistency, and strategies. A typical application scenario of translation is device migration. For instance, a customer intends to decommission an old equipment from vendor B and replace it with a new device from vendor A. However, the two vendors may support different capabilities to deliver the same network services. This replacement currently leads to the preparation of new vendor- specific configuration files, at the risk of configuration errors. NetGraph provides the capability of translating vendor-specific configuration files based upon vendor-agnostic data models to relieve network managers from potentially complex manually declarative tasks.
Inventory search: NetGraph provides a fine-grained search function across the whole network inventory information. The digital twin models in NetGraph are stored in a database as a graph. Users can issue a query for an object such as a physical or a logical link, and NetGraph will return the appropriate information. For instance, if a user wants to know whether there exists a link between two physical interfaces, he/she provides the IP addresses of the two interfaces as input and NetGraph will return the link information.
Network elements validation: NetGraph can assist the validation of network elements and relieve network managers from possibly heavy workload. For instance, if there is a layer 2 network loop, typical fault management operation consists in checking the MAC address entries of the relevant forwarding information bases and identifying the faulty devices. With NetGraph, all the network elements are abstracted into unified models and data is organized into graphs [11]. The network managers can easily discover the loop by validating the typologies of the relevant devices.

3 NETGRAPH: DESIGN DETAILS

3.1 Architecture

The NetGraph architecture is depicted in Figure 1. It includes three models (device, network, and service models) and a network service layer providing multiple built-in services such as the location service, the routing service, etc. The NetGraph platform is connected to physical devices through an interface with the data plane of the devices. Configuration files, YANG data [2] and other information related to device/network status are exchanged in the NetGraph via multiple interfaces or protocols. NetGraph connects to the application scenario layer via the network service layer, which supports network planning, automatic deployment, network maintenance, network optimization, etc.

3.2 Model Definition and Generation

The device model contains the description of functions embedded in devices and which is derived from configuration files (AAA, ACL,

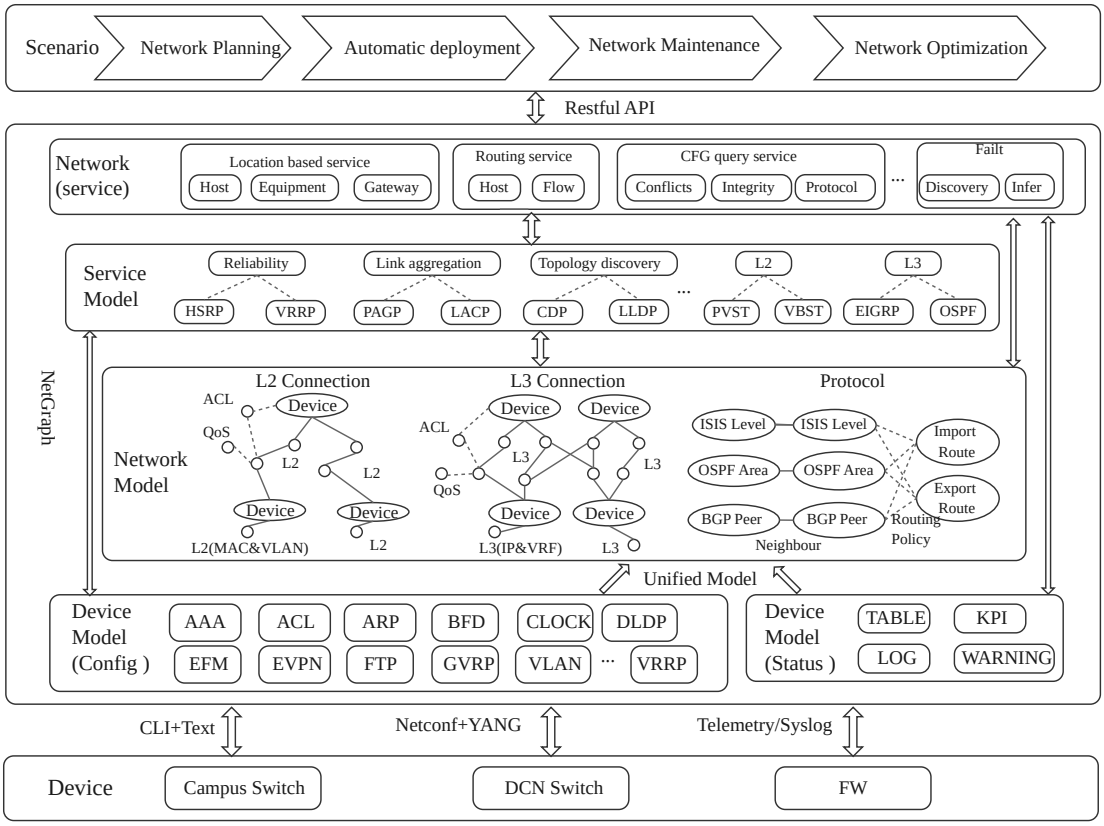

Figure 1: Architecture of NetGraph.

etc.) as well as other sources of information, which are used to describe the state of the network (key performance indicator, log, etc.). The network model depicts the topology of devices as well as routing adjacencies (ISIS, OSPF, etc.). It also abstracts the flow characteristics that correspond to the traffic forwarded within the network. The service model is an abstraction of various functions, which a network can provide, such as link aggregation or VPN. The above three models are generated progressively. We use the "entity-attribute-relation" mechanism [13] to represent the models in NetGraph. This mechanism uses the notion of "entity", which corresponds to a physical, independent object (interface, port, etc.) or a functional module (VRRP, ACL, etc.). The "Attribute" information describes the properties of an entity. For instance, the entity "interface" has "MAC address" and "interface number" as attributes. "Relation" illustrates the relationship between two entities, either physical or logical. To generate a device model, the original data are collected from different devices via multiple interfaces such as CLI, NETCONF, or telemetry. These data might be heterogeneous; it may be composed configuration files, YANG models, and network state information. When the data from a device have been collected and aggregated, entities and their corresponding attributes can be derived and the device model is elaborated accordingly. The network model describes the topology and connection policies of devices. Therefore, it is essential for NetGraph to acquire the information that pertains to the potential relationships between various devices. NetGraph defines multiple types of relationships such as "set", "containing", "neighbor", etc. The relationships are obtained by determining whether the entities hosted in different devices share one or several common attributes. For instance, both devices A and B include the "BGP" entity, and the "BGP" object owns the same attribute "peer Router ID", then the two devices are BGP routers that have established a BGP session. Physical relationships can be acquired utilizing a protocol such as Link Layer Discovery Protocol (LLDP [15]). When all the relationships between different devices have been acquired, the network model is elaborated accordingly. The service model is a higher abstraction of network functions and services. To generate the service model, a network manager needs to assemble the entities, attributes and relationships referring to a service. For instance, a Spanning Tree Protocol (STP) service comprises "bpdu filter", "stp process id", "interface", etc.

3.3 Configuration-Model Mapping

NetGraph collects various vendor-specific configuration templates and supports "configuration-model" bidirectional mapping. Figure 2 shows an example of a configuration file for an interface.

After receiving a configuration file, NetGraph derives the key information according to the model template (figure 3) and obtains the device model in figure 4.

```
interface 10GE1/0/1
    description to\_cneast2a-MGS-PSZ013-CNA012\_eth1
    port link-type trunk
    undo port trunk allow-pass vlan 1
    port trunk allow-pass vlan 574
    device transceiver 10GBASE-FIBER
```

Figure 2: An example of a configuration file.

```
    - regex: '^ port link-type (.*) $'
    examples:
    - ' port link-type trunk'
    captures:
    - link_type
    path:
    interface:
      "\"{{ parent_name }}\"":
      "{{ name }}":
      link_type: "\"{{ link_type }}\""
```

Figure 3: An example of model template

```
    interface_physical:
      10GE1/0/1:
        fwd_type: "l2"
        if_name: "10GE1/0/1"
        if_number: "1/0/1"
        phy_type: "10GE"
        slot_id: "1"
        vrf_name: "_public_"
        transceiver: "10GBASE-FIBER"
        description: "to_cneast2a-MGS-PSZ013-CNA012_eth1"
        link_type: "trunk"
        allow_vlan1: false
        allow_vlans: ["574"]
```

Figure 4: An example of a device model.

When a network manager intends to map a device model to a vendor-specific configuration file, he/she selects the appropriate template and the conversion process is executed automatically.

3.4 Model Management

NetGraph provides a flexible and fine-grained model management lifecycle. Since the device and network models are derived from data stored in configuration files as well as network status information, whenever these data are modified, the corresponding models will also change accordingly.

Models can be updated all at once or incrementally. In the former case, the network enters into a new lifecycle, all the changes of configuration and network status data need to be retrieved and parsed. The old data will be erased and the models will be reset to the initial configuration.

In the incremental mode, NetGraph analyzes the differences of data between two consecutive periods and refreshes the altered sections compared to the previous model. NetGraph records all the operations associated with models by creating a log file with a timestamp. When a data change happens, the operation with the newest timestamp will also be refreshed. Furthermore, NetGraph supports secure rollback. Whenever network failures occur due to some unexpected factors, NetGraph will back off to the latest version of data models to maintain its resilience and availability.

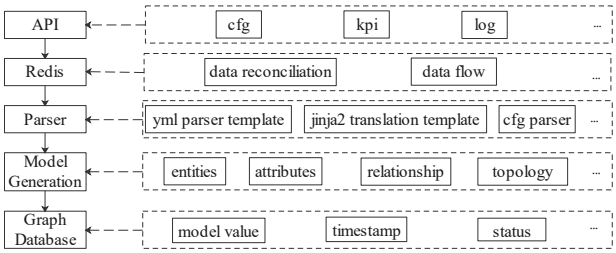

Figure 5: Model generation and storage procedure.

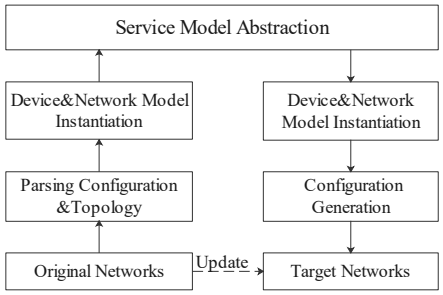

Figure 6: A translation process of network updating.

4 DEPLOYMENT

NetGraph has been deployed in Huawei data center networks for months and has been able to manage more than 50,000 devices. The whole system contains 20,000 lines of Python code and 80,000 lines of modular templates. In this section, we present the implementation of NetGraph in a real network environment. Specifically, we highlight some key challenges that have been addressed by our implementation.

4.1 Model Generation and Storage

NetGraph uses various techniques and hardware modules to realize its functional elements. An illustration of model generation and storage procedures of NetGraph is presented in Figure 5. The generation of data models relies upon the following steps:

Step 1: NetGraph collects the initial network data (configuration and maintenance) and transmit these data to the Redis.

Step 2: NetGraph parses the various vendor-specific templates and the network state data.

4.2 Network Change and Translation

NetGraph provides a translation when devices and networks need to be upgraded. To complete this deployment, NetGraph abstracts the service model from previous networks and conducts model translating tasks to generate the configuration files for devices in new networks. Specifically, a network update between different versions follows the steps illustrated in Figure 6.

Firstly, NetGraph parses the topology and configuration file obtained from original networks and instantiates them to generate device and network data models. Secondly, NetGraph abstracts the instantiated device and network data models to obtain the service

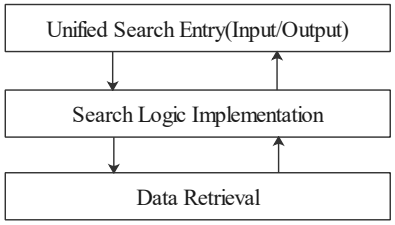

Figure 7: Three modules in inventory search.

data models. This step involves the modular templates of configuration files from different vendors. Lastly, NetGraph transforms the service data models into device and network data models. Specifically, it generates vendor-specific configuration files and forwards them to the relevant devices of the target networks.

4.3 Inventory Search

NetGraph provides a translation when devices and networks need NetGraph's Inventory search consists of three functional modules that are illustrated in Figure 7. The unified search entry module resides on top of the structure; it handles users' search requests and displays the results in a web format. The search logic implementation layer executes Inventory search functions. It transmits the search input through the internal query interface and processes the search requests. When these tasks are completed, the search logic module returns the output to a unified search entry.

The data retrieval module interacts with the graph retrieval database and extracts the required information according to search requests delivered by the internal query interface. The network path search is a typical application of NetGraph's search function. When a user sends a request to query the link path between two IP addresses, NetGraph acts as follows:

Step 1: Unified search entry captures user's input and sends the IP strings to the search module.

Step 2: The search logic transmits the IP address strings into query messages sent to the data retrieval module.

Step 3: The data retrieval module conducts search operations over the graph retrieval database and returns the entities along with the link path containing the target IP addresses. Other information such as attributes, maintenance data relevant to the entities in the link path is also included in the search results.

Step 4: Unified entry will display the results to users via a web interface.

5 PERFORMANCE EVALUATION

5.1 Network Scale and Model Coverage

Statistical data show that NetGraph is used to manage more than 50,000 devices while data models contain more than 10 million entities, 100 million attributes along with 20 million connections established within the network. Besides, NetGraph is capable of providing 50 million rows of CLI commands, which supports a wide range of configuration templates for most mainstream vendors.

In order to provide secure auditing and rollback functions, NetGraph records all the alterations of entities in device and network

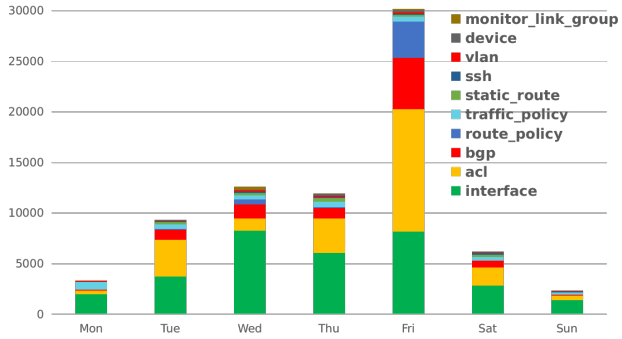

Figure 8: Alterations during one-week period.

data models. The total memory size required to store these data models is 6.5 GB. Figure 8 lists the top 10 entities where alterations occur during one week. NetGraph creates and stores the log files with timestamps to record these alterations.

5.2 Time Consumption

We evaluated NetGraph's modeling capabilities with regard to the time consumption of the three basic function modules. The following procedures are denoted:

Parse: NetGraph takes configuration files and other network maintenance relevant data as input and outputs the models. Translate: NetGraph maps the device and network data models to the corresponding configuration files.

Graph: NetGraph constructs a global graph comprising all the entities, attributes and relationships.

NetGraph's procedures are deployed on four distributed servers running Linux 3.10.0; each server has a 64GB memory with 12 Intel(R) 2.2 GHz Core CPU. The total time consumption required to parse all the devices is 54.26 minutes, while it takes 53.04 minutes and 22.5 minutes respectively for NetGraph to complete translate and graph procedures. Figure 9 illustrates the total time consumption of NetGraph's procedures.

Figure 10 illustrates the time consumption variation of the three function modules as a function of the number of devices. In general, the time consumed for NetGraph processing increases with the number of devices. In addition, we evaluated the time consumption of inventory search over source/destination pairs with different hops and the result is displayed in Figure 11. We divided the scenarios by the hops between source and destination IP addresses, and conducted the search procedure twenty times for each scenario. It takes 31.8ms, 62.5ms and 119.4ms respectively when the hops between the two IP addresses are one, two and three, respectively. Processing times increases with the number of hops. To sum up, the processing times of these procedures remain attractive while

6 DISCUSSION

6.1 Limitations

Scenario coverage: NetGraph supports the capability of bidirectional mapping between data models and configuration files in Huawei data center networks. However, new challenges arise when

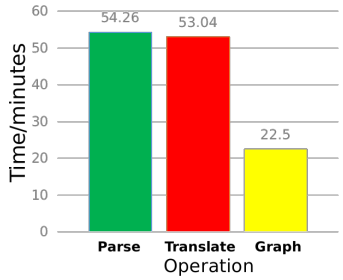

Figure 9: Total time consumption of NetGraph's procedures.

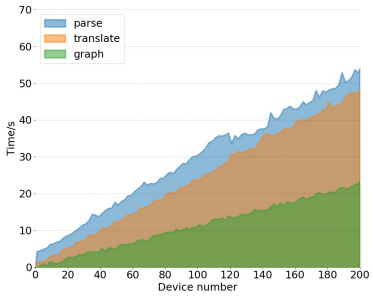

Figure 10: Time consumption with the changing amount of devices.

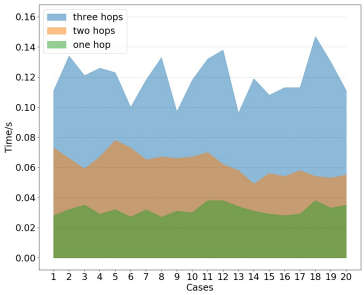

Figure 11: Time consumption of search over IP pairs.

we attempt to apply NetGraph to other network environments. Firstly, network elements in different scenarios can be of various technologies. As a consequence, some entities have not been included in NetGraph so far. For instance, the campus scenario includes elements "access point" which do not appear in a DCN scenario. Furthermore, new network scenarios and architectures may be composed of devices from different vendors, which are provisioned by new configuration templates. NetGraph will fail to complete modeling or translating tasks when a new scenario is applied for the first time.

Results ranking of inventory search: NetGraph can display all the physical paths, logical paths, and significant status information about the network elements. The amount of data also leads to another problem: users may find it difficult to extract useful information from the tremendous amount of information provided by the results of the search. NetGraph is expected to optimize the results ranking algorithm and users' desired targets can be obtained from the leading search outcomes.

6.2 Future work

In addition to data center networks, NetGraph will iterate consistently to be suited for additional network scenarios such as WAN, campus, etc. To fulfill this purpose, we have to get acquainted with the characteristics of network facilities in new scenarios and integrate more configuration templates from mainstream vendors.

Besides, we will optimize the NetGraph's search module in terms of ranking via multiple mechanisms. NetGraph will analyze the occurrence frequency of each entity being searched, and adjust the strategy of ranking according to users' feedbacks or requests. For instance, NetGraph can record users' selections after the search results are displayed. If a certain item is assigned a higher priority than others by users, NetGraph will rank this item at the forefront of results when the same keyword is typed in the follow-up search.

7 RELATED WORK

Many large enterprises have devoted their efforts to network management. Facebook [6] presents Robotron to realize top- down network management. Robotron allows the network managers to represent their intents and the corresponding low-level design will be generated automatically. Robotron has been deployed in Facebook networks and achieves good performances. Alibaba [7] proposes NetCraft to achieve automatic lifecycle management of network configuration. NetCraft abstracts the network into a multi-layered graph model and can automate the configuration lifecycle. Microsoft [8] presents Statesman, which uses three views to describe the network status. It allows each application in the network to run independently while keeping the network-wide invariants unaltered. Statesman has been deployed in ten Microsoft Azure datacenters for months with three different applications. Google [9] proposes MALT, a multi-abstraction layer topology representation to describe the network. MALT represents the network using the "entity-relationship model" and divides Google's entire network into multiple shards. Network managers can use MALT to execute all network management processes, such as design, configuration and deployment.

The main differences between these works and NetGraph are from the following three aspects: Firstly, we build a full virtual image of the physical network, which comprises the entire network elements including configuration files and state information, thus constructing an integrated digital twin platform towards automatic and intelligent DCN management. Secondly, NetGraph supports bidirectional mapping and this property has laid a solid foundation for smart network service migration. Thirdly, we have developed multiple functional blocks such as inventory search to enrich the application scenarios of NetGraph.

8 CONCLUSION

In this paper, we presented NetGraph, an intelligent operated digital twin platform for Huawei's data center networks. NetGraph tackles the network management challenges by abstracting the network into three models and integrating four functional blocks. The real network statistics demonstrate the performance of NetGraph. Our future research directions will focus on the development of additional application scenarios and the functional optimization of NetGraph.

ACKNOWLEDGMENTS

We thank our colleagues Tengxiang Zhang, Zhenwei Zhang, Xiaoyang Xu, Yanping Cao, Qiang Yang for their contributions to NetGraph. We also acknowledge Jacquenet Christian, Mohamed Boucadair, our shepherd Prof Yong Cui and other reviewers for their helpful comments and suggestions on our manuscript.

REFERENCES

[1] Huawei DCN Applications, https://info.support.huawei.com/network/ptmngsys/Web/WDMkg/en/27_dcn.html

[2] M. Bjorklund. YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF). RFC 6020

[3] W. Enck et al. Configuration management at massive scale: system design and experience. IEEE Journal on Selected Areas in Communications, 2009.

[4] H. H. Liu, X. Wu, M. Zhang, L. Yuan, R. Wattenhofer, and D. Malt. zUpdate: Updating data center networks with zero loss. In ACM SIGCOMM Computer Communication Review, volume 43, pages 411–422. ACM, 2013.

[5] Justin Meza, Tianyin Xu, Kaushik Veeraraghavan, and Onur Mutlu. 2018. A Large Scale Study of Data Center Network Reliability. In Proceedings of the Internet Measurement Conference 2018 (IMC '18). Association for Computing Machinery, New York, NY, USA, 393–407.

[6] Yu-Wei Eric Sung, Xiaozheng Tie, Starsky H.Y. Wong, and Hongyi Zeng. 2016. Robotron: Top-down Network Management at Facebook Scale. In Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM '16). Association for Computing Machinery, New York, NY, USA, 426-439.

[7] Hongqiang Harry Liu, Xin Wu, Wei Zhou, Weiguo Chen, Tao Wang, Hui Xu, Lei Zhou, Qing Ma, and Ming Zhang. Automatic Life Cycle Management of Network Configurations. In Proceedings of the Afternoon Workshop on Self-Driving Networks (SelfDN 2018). Association for Computing Machinery, New York, NY, USA, 29-35.

[8] Peng Sun, Ratul Mahajan, Jennifer Rexford, Lihua Yuan, Ming Zhang, and AhsanArefin. 2014. A network-state management service. In Proceedings of the 2014 ACM conference on SIGCOMM (SIGCOMM '14). Association for Computing Machinery, New York, NY, USA, 563-574.

[9] Mogul J C, Goricanec D, Pool M, et al. Experiences with modeling network. Networked Systems Design and Implementation. NSDI 20. 2020: 403-418.

[10] Xu Chen, Yun Mao, Z. Morley Mao, and Jacobus Van der Merwe. Declarative Configuration Management for Complex and Dynamic Networks. In Proc. CoNEXT, 2010.

[11] Grzegorz Malewicz, Matthew H. Austern, Aart J.C Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz, Czajkowski. Pregel: A System for Large-scale Graph Processing. In Proc. SIGMOD, pages 135–146, 2010.

[12] OpenConfig. http://www.openconfig.net/.

[13] Peter Pin-Shan Chen. The Entity-relationship Model-Toward a Unified View of Data. ACM Trans. Database Syst., 1(1):9–36, March 1976.

[14] Grieves M. Digital twin: Manufacturing excellence through virtual factory replication [Online], available: https://www.3ds.com/ fileadmin/PRODUCTS-SERVICES/DELMIA/PDF/Whitepaper/DELMIA-APRISO-Digital-Twin-Whitepaper.pdf, March 11,2021.

[15] 802.1AB Overview Link Layer Discovery Protocol, https://www.ieee802.org/3/frame_study/0409/blatherwick_1_0409.pdf

[16] C. Zhou, H. Yang, X. Duan, et al. Concepts of Digital Twin Network. Internet-Draft: draft-zhou-nmrg-digitaltwin-network-concepts-03, Feb 22, 2021

[17] Sun Tao, Zhou Cheng, Duan Xiao-Dong, et al. Digital twin network (DTN): concepts, architecture, and key technologies. Acta Automatica Sinica, 2021, 47(3): 569-582.