Variable-Granularity Digital Twin Construction Techniques for Software-Defined Networking: Postprint

Authors: Chen Hexiong, Wu Jiaping, Wei Yunkai, Guo Wei, Hang Feilu, Mao Zhengxiong, Yang Ning

Date: 2022-05-18T00:00:00+00:00

Abstract

The expanding application scope of Software Defined Network (SDN) brings challenges of diversified application requirements. Utilizing Digital Twin (DT) to enhance SDN's real-time analysis, deduction, and control capabilities can better meet the requirements of various application scenarios. However, the current construction of digital twins for SDN faces problems of high latency requirements, large computational overhead, and difficult resource coordination. Therefore, oriented by application requirements and under the constraints of network-available computing resources, a novel concept of Variable Granularity Digital Twin (VGDT) and its construction technology is proposed. VGDT combines the distribution characteristics of network-available computing resources to establish a multi-node resource collaborative optimization model that guarantees digital twin latency and completeness. On this basis, a hybrid encoding genetic algorithm is used to solve the model, obtaining the optimal mapping data granularity and digital twin deployment scheme to guide the construction process of the digital twin. Simulation results show that, compared with existing models, under network computing resource constraints, VGDT has higher digital twin model completeness and model effectiveness.

Full Text

Variable Granularity Digital Twin Construction Technology for Software Defined Networks

Chen Hexiong[1], **Wu Jiaping**[2],[3], **Wei Yunkai**[2],[3]†, **Guo Wei**[1], **Hang Feilu**[1], **Mao Zhengxiong**[1], **Yang Ning**[3] [1]. Information Center of Yunnan Power Grid Co. Ltd., Kunming 650000, China;
[2]. Yangtze Delta Region Institute (Quzhou), University of Electronic Science &

Technology of China, Quzhou 324000, China;
[3]. School of Information & Communication Engineering, University of Electronic Science & Technology of China, Chengdu 611731, China

Abstract: The expanding application scope of Software Defined Networks (SDN) presents challenges in meeting diverse application requirements. Digital Twin (DT) technology can enhance SDN's capabilities in real-time analysis, deduction, and control, thereby better satisfying various application scenario demands. However, current DT construction for SDN faces significant challenges including strict delay requirements, heavy computational overhead, and difficult resource coordination. Therefore, this paper proposes a novel Variable Granularity Digital Twin (VGDT) concept and its construction technology, guided by application requirements and constrained by available network computing resources. VGDT establishes a multi-node resource collaborative optimization model that guarantees DT delay and completeness by leveraging the distribution characteristics of available network computing resources. Furthermore, a hybrid coding genetic algorithm is employed to solve this model, obtaining optimal mapping data granularity and DT deployment schemes to guide the DT construction process.

Simulation results demonstrate that compared with existing approaches, VGDT achieves higher DT model integrity and validity under network computing resource constraints.

Keywords: Software Defined Network; Digital Twin; Variable Granularity; Genetic Algorithm; Application Driven

0 Introduction

With the development of information and communication technologies, Software Defined Network (SDN) applications have become increasingly widespread, emerging in novel scenarios such as information-based industrial plants [1], power grids [2], and digital healthcare [3]. These diverse application scenarios impose increasingly varied and complex demands on SDN, requiring more effective organization, management, and scheduling of network resources to meet dynamic application requirements [4]. Digital Twin (DT) technology provides a promising solution to efficiently address these needs. By constructing real-time digital mirrors of physical entities, DT enables monitoring, simulation, and control of physical entities, and is being extensively researched and applied in manufacturing, healthcare, smart cities, and other domains [5]. DT-empowered SDN can enhance network perception, analysis, and control capabilities, thereby more effectively organizing, managing, and scheduling network resources to meet dynamic demands across various application scenarios. DT construction for SDN is both the prerequisite and foundation for fulfilling these requirements and represents a current research hotspot.

Current DT construction approaches for SDN can be categorized into two types: independent construction and integrated construction. Independent construc-

tion schemes build DT on completely separate infrastructure (servers or networks). For instance, Zhao et al. [6] proposed constructing DT for Software Defined Vehicular Networks (SDVN) on centralized servers or controllers, while Krishnan et al. [7] independently constructed DT for Software Defined Internet of Things (SD-IoT) on a central server. In independent construction schemes, the substantial communication overhead between DT construction facilities and the forwarding plane increases communication delay for DT-network interactions. Moreover, when analyzing SDN, these facilities must simultaneously process data from the entire network, creating heavy computational burdens and requiring additional network infrastructure overhead. Consequently, for DT with strict delay requirements [8, 9], this construction approach often yields unsatisfactory results and incurs significant communication and computational costs.

Integrated construction schemes, by contrast, utilize existing SDN facilities (such as edge servers) to collaboratively construct DT for the physical network, integrating DT construction into the target physical network. Khan et al. [10] proposed a DT-based 6G network architecture, comparing cloud-based independent DT construction with edge-based DT construction, with the latter offering lower communication latency and higher scalability. Xu et al. [11] achieved integration of vehicular edge computing and digital twins by establishing digital twins for individual vehicles on multiple Roadside Units (RSUs). Zhang et al. [12] constructed digital twins for vehicles in Vehicular Edge Computing Networks (VECN) on edge servers. In integrated construction schemes, edge servers distributed throughout the physical network enable shorter communication distances to network devices, reducing DT communication latency. Additionally, through collaboration among multiple edge servers, the computational burden of DT is distributed across various edge servers, improving computational efficiency. This approach is generally preferred when strict requirements exist for DT construction delay and completeness.

However, existing integrated DT construction approaches typically aim to establish complete mappings of the physical network, striving for 100% fidelity between the DT and physical network in virtual space. This complete mapping approach often results in substantial resource overhead and system pressure. Lopez et al. [13] noted that when applying DT for context-aware data processing and behavioral simulation in smart grids, this approach requires enormous communication and computational resources. In practice, complete mapping DT is not necessarily required under specific application scenarios and their demand characteristics. Yang et al. [14] proposed a DT-based network traffic simulation framework that improves traffic prediction efficiency through traffic sampling to reduce data transmission volume from network to DT. While this approach alleviates DT construction pressure to some extent, its application mode is relatively fixed and singular.

Addressing current challenges in DT construction, this paper proposes a Variable Granularity Digital Twin (VGDT) concept and its construction technology.

Under limited network computing resources, VGDT adjusts mapping granularity according to application characteristics to maximize DT model effectiveness and meet application requirements.

The main contributions of this paper are as follows:

1) We propose a variable granularity digital twin concept and construction architecture for SDN that leverages available computing resources in the network to build DT meeting application requirements.
2) We establish a variable granularity digital twin effectiveness optimization model that ensures DT model construction delay, completeness, and validity by adjusting mapping granularity and twin placement, thereby providing DT that satisfies application demands.
3) We validate VGDT effectiveness through simulation, demonstrating that under computing resource constraints, VGDT achieves higher DT model completeness and validity, better meeting application requirements.

1 Related Fundamentals

This section briefly introduces Software Defined Networks and Digital Twin technology to provide a foundation for the subsequent proposal of variable granularity digital twin architecture and techniques.

1.1 Software Defined Network

As a novel network architecture attracting significant attention in recent years, SDN enables more flexible deployment of network functions through programmable network management, improving network application performance and management efficiency. The 2016 SDN white paper published by the Open Networking Foundation specifies that SDN consists of an application plane, control plane, and data plane. The control plane connects upward to the network application plane via northbound interfaces and downward to the data plane via southbound interfaces, enabling programmable network management according to application plane requirements and unified control of data plane forwarding using network-wide data, thereby allowing SDN to programmatically manage networks based on application demands and improve SDN application performance [15].

SDN provides a rich and diverse data foundation for controllers to formulate network management strategies and deploy network functions by collecting network-wide data in real-time. Moreover, SDN's centralized control through analysis and utilization of network-wide data facilitates global network management and optimization. However, as SDN application scenarios diversify and scale expands, the volume of network data gradually increases, imposing higher demands on the control plane's ability to fully collect, process, and utilize network data.

1.2 Digital Twin

The concept of Digital Twin first emerged in 2003 when Grieves introduced it into product lifecycle management courses. In 2012, the U.S. Air Force Research Laboratory and NASA formally proposed the "Digital Twin" concept and applied DT to aerospace vehicle design and maintenance. Since then, with rapid advances in computer technology, multidisciplinary modeling, and simulation technology, DT has been widely applied in manufacturing, healthcare, smart cities, and other fields [16]. Current research and applications demonstrate that DT is an intelligent, continuously evolving system that establishes virtual twins of physical world entities in virtual space, utilizing big data, artificial intelligence, and other technologies to model, simulate, and analyze the state and behavior of physical entities, as well as predict and control their states and behaviors [8, 17]. These physical entities can be complex physical systems such as equipment, robots, workshops, or communication networks, while virtual twins are multi-granularity mappings of physical entities. The primary goal of DT is to establish high-fidelity mapping models of physical entities that truthfully reflect all information about physical entities in virtual space, guiding physical entity operation through analysis of virtual twin states and behaviors to achieve collaborative evolution between physical entities and virtual twins.

Many studies have already applied DT to physical networks. References [7, 10, 12] applied DT to industrial IoT, vehicular networks, and edge networks, demonstrating DT's strong potential for improving communication network application performance. Integrated DT construction for physical networks builds digital twins for each device on computing nodes closer to the network devices, featuring low interaction latency and reduced network costs. However, this approach faces challenges when network locations and computing resources of these DT construction nodes in the physical network change, as it requires integrating distributed computing resources across the network.

2 Problem Analysis and System Architecture

This section first analyzes the resource limitation challenges faced in integrated DT construction within SDN networks, and subsequently proposes the concept of variable granularity digital twin. Based on this, we present the system architecture for constructing variable granularity digital twins in SDN.

2.1 Problem Analysis

Constructing DT for SDN can enhance SDN's perception, analysis, and control capabilities for networks, helping to improve the processing capabilities for complex network data in application scenarios. Simultaneously, when constructing DT in an integrated manner within SDN, SDN's global perspective and centralized network control characteristics facilitate efficient utilization of distributed computing resources across the network for DT construction. Therefore, integrated DT construction for SDN not only helps SDN more efficiently

meet application scenario demands but also fully utilizes SDN network resources to improve resource coordination capabilities among computing devices during DT construction with lower latency. However, traditional DT construction approaches with complete mapping mechanisms impose substantial computational overhead on the network. Consider the following two scenarios:

1) When DT is used to improve SDN's network security defense capabilities, complete mapping-based DT can identify potential risks at fine granularity under high network risk conditions, enabling proactive measures. However, when network attack risks are low, both complete mapping and appropriately reduced-granularity mapping can achieve equivalent security defense capabilities [15, 19]. In such cases, mapping granularity can be appropriately reduced to decrease resource overhead while maintaining the same security defense capability.

2) When DT assists SDN in network resource scheduling, complete mapping-based DT can analyze network status and schedule resources more precisely through detailed data analysis for managing current or short-term network resources. However, when managing network resources over longer time spans, both complete mapping and appropriately reduced-granularity mapping can achieve equivalent resource scheduling effects, with the latter incurring lower network resource overhead.

Therefore, complete mapping digital twins of SDN are not necessarily required under practical application scenario demands. This paper proposes the variable granularity digital twin concept, which constructs more effective DT by adjusting the mapping granularity of physical networks in DT and the placement of twins based on application scenario demand characteristics and available computing resources in the network.

To achieve "variable granularity," VGDT can adjust granularity along two dimensions: data update cycle and data volume per cycle.

As shown in Figure 1: see original paper, adjusting the data upload cycle from each device can change the data update cycle of each twin in DT. If DT needs to compute and analyze historical network data, such as using Recurrent Neural Network (RNN) to analyze network traffic, reducing the length of input sequential data can decrease neuron iteration counts, thereby reducing computation time. As shown in Figure 1: see original paper, adjusting the data volume updated per cycle for each device twin not only reduces communication volume during each data upload but also changes DT's data analysis and processing time. For instance, when DT needs to perform dimensionality reduction on matrices composed of network data using Principal Component Analysis (PCA), reducing the data volume of each input matrix can lower PCA computational complexity, thereby saving DT data processing time.

Both granularity adjustment methods achieve the goal of reducing DT construction resource overhead. Although these two methods differ in form, they essentially reduce data update requirements in each DT construction cycle. There-

fore, the "variable granularity" discussed in this paper can refer to variable cycles or variable data granularity under fixed cycles. For narrative convenience, this paper uniformly represents "variable granularity" as data granularity adjustment under fixed cycles.

2.2 Variable Granularity Digital Twin System Architecture

[Figure 2: see original paper] illustrates the variable granularity digital twin construction technology architecture for SDN. This architecture employs an integrated construction approach that leverages distributed available computing resources in SDN (such as edge servers and user devices with surplus computing resources after completing inherent tasks, collectively referred to as edge servers for convenience) to construct variable granularity digital twins for SDN. In VGDT, the controller or edge servers connected to the controller analyze the impact of mapping granularity for various elements and twin placement positions on DT construction delay and model completeness based on application scenario demand information and available computing resources on each edge server, thereby formulating optimal DT construction schemes for the network. Subsequently, the controller or connected edge servers broadcast DT construction information across the network, including elements required to fulfill application demands, mapping granularity for each element, and twin placement positions for each device. Hosts and switches then collect, process, and transmit mapping data required by application scenarios to target edge servers according to the mapping granularity information, establishing local network DT on each edge server. Finally, the controller or connected edge servers communicate and interact with other edge servers to obtain and synchronize information from each local network DT, connecting local network DTs to establish the entire network's DT, thereby achieving integrated DT construction in SDN.

Assume SDN contains N_s switches forming set $S = \{s_1, s_2, ..., s_{N_s}\}$, with each switch s_i (where $1 \leq i \leq N_s$) connected to hosts h_{ij} (where $1 \leq j \leq n_i$). These hosts form set $H = \{h_1, h_2, ..., h_{N_h}\}$. Each edge server e_k with surplus computing resources in the network (where $1 \leq k \leq L$) forms set $E = \{e_1, e_2, ..., e_L\}$, where L represents the number of edge servers. Let $f_t(k)$ denote the available computing resources on e_k, representing the equivalent number of computing cycles available for DT construction. All available computing resources in SDN form set $F_t = \{f_t(1), f_t(2), ..., f_t(L)\}$. Since edge servers are distributed throughout the network, if host h_{ij} establishes its twin on e_k, we have $l_{ij}^k = 1$; otherwise $l_{ij}^k = 0$. For simplicity, if h_i establishes its twin on e_k, we have $l_i^k = 1$; otherwise $l_i^k = 0$.

In DT constructed for SDN, digital twins of switches and hosts can be uniformly represented as $\langle Data_t, State_t, Data_{t+1} \rangle$, where $Data_t$ represents historical data mapped from device i to DT according to application demands, $State_t$ represents the twin's operational state when fulfilling application demands, and $Data_{t+1}$ represents data mapped from device i to its twin in the next cycle. For historical data possessed by twins, $Data_t$ includes static data characteriz-

ing device physical features, such as CPU frequency, memory size, and physical connection relationships of s_i and h_j, as well as dynamic data reflecting device operational status, such as forwarding flow tables issued by controllers on s_i, packet forwarding logs, and communication buffer sizes and network attack logs of h_j. To fulfill application demands, switches in SDN map N_s types of elements to VGDT, with each element's mapping granularity represented as ω_{s_i} (where $0 < \omega_{s_i} \leq 1$). These mapping granularities form set $\Omega_s = \{\omega_{s_1}, \omega_{s_2}, ..., \omega_{s_{N_s}}\}$. Similarly, hosts in SDN map N_h types of elements to VGDT, with these elements' mapping granularities represented as ω_{h_j} (where $0 < \omega_{h_j} \leq 1$) and forming set $\Omega_h = \{\omega_{h_1}, \omega_{h_2}, ..., \omega_{h_{N_h}}\}$.

3 Variable Granularity Digital Twin Optimization Model

When constructing DT for SDN based on the variable granularity concept, application demands for VGDT are represented as a triple $\langle I_0, D_0, E_0 \rangle$, where I_0 is model completeness, representing the ratio between DT constructed based on variable granularity and DT based on complete mapping; D_0 is model construction delay, representing the time required from network data generation to DT update completion by VGDT based on new data; E_0 is model effectiveness, a comprehensive measure of model timeliness and completeness, representing the degree to which VGDT meets application demands under available computing resource constraints. This section establishes the variable granularity digital twin optimization model based on analysis of model construction delay, completeness, and effectiveness.

3.1 Granularity Analysis

Driven by application demands, VGDT adjusts mapping granularity for various elements and twin placement positions of switches and hosts in DT according to available computing resources in the network, thereby affecting model completeness I_t and model construction delay D_t during DT construction. I_t is primarily related to mapping granularity of various elements, with higher mapping granularity yielding higher I_t. This paper uses function $\psi(\Omega_s, \Omega_h)$ to measure current I_t:

$$I_t = \psi(\Omega_s, \Omega_h) = \sum_{i=1}^{N_s} w_{s_i} \log_2(1 + \omega_{s_i}) + \sum_{j=1}^{N_h} w_{h_j} \log_2(1 + \omega_{h_j})$$

where w_{s_i} and w_{h_j} are importance weights of the i-th switch element and j-th host element for DT model completeness, with $\sum_{i=1}^{N_s} w_{s_i} + \sum_{j=1}^{N_h} w_{h_j} = 1$ and $0 < w_{s_i}, w_{h_j} < 1$.

Since available computing resources in SDN are distributed, changes in mapping granularity and twin placement positions for hosts and switches affect communication and computational delays in DT construction. This paper defines model

construction delay D_t as the average model construction delay across these edge servers:

$$D_t = \frac{1}{L}\sum_{k=1}^{L}\left(Cal_k(D_t) + Com_k(D_t)\right)$$

where $Cal_k(D_t)$ and $Com_k(D_t)$ are computational delay and communication delay for constructing DT on edge server e_k, respectively.

During DT construction, the data volumes transmitted by s_i and h_j to target e_k for DT updates are φ_{ij}^t and φ_i^t, respectively. Propagation delay t_{pg} represents the time for electromagnetic wave transmission, calculated as $t_{pg} = \frac{d}{c}$, where d is communication distance and c is light speed in the medium [21].

Computational delay $Cal_k(D_t)$ represents the sum of computing cycles required to build all twins on e_k, which depends on data volume and mapping granularity of twin elements. This paper uses $\theta(\cdot)$ to measure computing cycles needed for each twin:

$$Cal_k(D_t) = \sum_{i=1}^{N_s} l_i^k \cdot \theta(\varphi_{ij}^t, \omega_{s_i}) + \sum_{j=1}^{N_h} l_{ij}^k \cdot \theta(\varphi_i^t, \omega_{h_j})$$

where:

$$\theta(\varphi_{ij}^t, \omega_{s_i}) = \theta_{s_i}^{\text{size}} \cdot \varphi_{ij}^t \cdot \omega_{s_i}, \quad \theta(\varphi_i^t, \omega_{h_j}) = \theta_{h_j}^{\text{size}} \cdot \varphi_i^t \cdot \omega_{h_j}$$

$\theta_{s_i}^{\text{size}}$ and $\theta_{h_j}^{\text{size}}$ represent computing cycles required for complete mapping of switch and host elements, respectively, i.e., the conversion ratio between computing cycles and data volume for building twins.

Communication delay $Com_k(D_t)$ primarily includes communication delay between hosts, switches, and edge servers, as well as interaction delay among edge servers for collaborative DT construction. During real-time interaction between hosts/switches and target edge servers, hosts and switches mainly transmit DT update data to edge servers, while edge servers primarily transmit control messages such as mapping granularity and twin construction positions to hosts and switches, with the latter communication volume being much smaller. Therefore, this paper considers communication delay between hosts/switches and edge servers mainly as mapping data update delay. Additionally, since the edge server collaboration scheme adopted in this paper is similar to other schemes [20], the interaction delay for DT construction shows no significant difference, so this paper primarily considers the delay for edge servers to receive twin update data as communication delay. The average communication delay for switches and hosts sending update data to e_k is:

$$Com_k(D_t) = \sum_{i=1}^{N_s} l_i^k \cdot Com_{s_i \to e_k} + \sum_{j=1}^{N_h} l_{ij}^k \cdot Com_{h_j \to e_k}$$

where $Com_{s_i \to e_k}$ and $Com_{h_j \to e_k}$ are communication delays for s_i and h_j sending DT update data to e_k, respectively.

In SDN, communication delay between two devices consists of four components: queuing delay, processing delay, transmission delay, and propagation delay. Considering that processing delay and queuing delay are typically related to specific application network characteristics and remain relatively constant within the same network, this paper simplifies analysis by adopting mean values while maintaining generality. The processing delay per hop uses mean t_{pr}, and queuing delay per hop uses mean t_{qu}. Transmission delay t_{tr} represents the time from sending the first bit to sending the last bit of a packet, i.e., $t_{tr} = \frac{B}{R}$, where B is packet data volume and R is switch data transmission rate [12].

Thus, the communication delay for s_i and h_j sending DT update data to e_k is calculated as:

$$Com_{s_i \to e_k} = t_{qu} + t_{pr} + t_{tr} + t_{pg} = t_{qu} + t_{pr} + \frac{B_{s_i}}{R} + \frac{d(s_i, e_k)}{c}$$

$$Com_{h_j \to e_k} = t_{qu} + t_{pr} + t_{tr} + t_{pg} = t_{qu} + t_{pr} + \frac{B_{h_j}}{R} + \frac{d(h_j, e_k)}{c}$$

where $d(s_i, e_k)$ and $d(h_j, e_k)$ are communication distances from s_i and h_j to target e_k, respectively.

3.2 Optimization Model

Under application demands, VGDT adjusts mapping granularity for various elements and digital twin placement positions for switches and hosts based on available computing resource distribution, thereby reducing data communication volume from switches and hosts to DT and average computation on edge servers, decreasing model construction delay and improving twin completeness to build more effective DT through full utilization of available network computing resources. Therefore, to provide more effective DT for applications, VGDT should jointly optimize mapping granularity for various DT features and twin placement positions based on existing network computing resources. This paper models DT model effectiveness E_t as:

$$E_t = I_t - \alpha \times D_t$$

where α is the impact factor of model construction delay on model effectiveness.

The optimization model is formulated as:

$$
\begin{aligned}
\arg \max_{\Omega_s, \Omega_h, Link} \quad & E_t = I_t - \alpha \times D_t \\
\text{s.t.} \quad & C_1 : I_0 \le I_t \le 1; \quad D_0 \ge D_t \ge 0; \quad E_0 \le E_t \le 1 \\
& C_2 : \omega_{s_i}^{\min} \le \omega_{s_i} \le 1, \quad 1 \le i \le N_s \\
& C_3 : \omega_{h_j}^{\min} \le \omega_{h_j} \le 1, \quad 1 \le j \le N_h \\
& C_4 : \sum_{k=1}^{L} l_i^k = 1, \quad 1 \le i \le N_s \\
& C_5 : \sum_{k=1}^{L} l_{ij}^k = 1, \quad 1 \le j \le n_i, \quad 1 \le i \le N_s \\
& C_6 : l_i^k, l_{ij}^k \in \{0, 1\}
\end{aligned}
$$

where $Link$ is the set of twin placement position variables l_i^k and l_{ij}^k, represented as:

$$
Link = \begin{bmatrix}
l_1^1 & l_1^2 & \cdots & l_1^L \\
l_2^1 & l_2^2 & \cdots & l_2^L \\
\vdots & \vdots & \ddots & \vdots \\
l_{N_s}^1 & l_{N_s}^2 & \cdots & l_{N_s}^L
\end{bmatrix}
$$

In the constraints, C_1 ensures that model effectiveness, completeness, and construction delay satisfy application requirements. C_2 and C_3 indicate that data volume for each mapped element of switches and hosts must meet minimum requirements for DT construction ($\omega_{s_i}^{\min}$ and $\omega_{h_j}^{\min}$). C_4 and C_5 specify that each switch and host can only establish its twin on one edge server. Since variables ω_{s_i}, ω_{h_j} in equation (10) are continuous and l_i^k, l_{ij}^k are discrete, this optimization model is a typical Mixed Integer Programming (MIP) problem, which is NP-Hard [22] and cannot be solved in polynomial time.

4 Hybrid Coding Genetic Algorithm

MIP is commonly used in production planning, resource allocation, and other problems. Current solution methods for MIP include branch and bound, Lagrangian relaxation, and heuristic algorithms. Branch and bound is often used for smaller-scale problems and can obtain optimal solutions. Lagrangian relaxation reduces problem constraints to obtain approximate solutions, but its solution scale is limited and optimal solutions are not guaranteed. Heuristic algorithms, proposed through observation of natural laws, can obtain feasible solutions in polynomial time and have been applied in many studies. Genetic algorithm, a typical heuristic algorithm, simulates biological inheritance and

evolution processes in natural environments, serving as a global adaptive probability search algorithm that gradually optimizes solutions. It is commonly used for solving NP-Hard and NP-Complete optimization problems [23]. In genetic algorithms, each individual in the population represents a candidate solution, and through multiple iterations of selection, crossover, and mutation, optimal solutions satisfying convergence conditions can be obtained.

Since variables ω_{s_i}, ω_{h_j} in equation (10) are continuous and l_i^k, l_{ij}^k are discrete, this paper performs segmented coding of genetic algorithm chromosomes: the first segment uses real-number coding for ω_{s_i} and ω_{h_j}, and the second segment uses binary coding for l_i^k and l_{ij}^k. When solving equation (10) with genetic algorithm, key parameters affecting solution efficiency include population size p_N, crossover probability p_c, mutation probability p_m, and iteration count d_N, with equation (10) serving as the fitness function. The following sections describe the selection, crossover, and mutation steps for solving equation (10) using the hybrid coding genetic algorithm.

4.1 Selection Operator

In genetic algorithm parent individual selection strategies, roulette wheel selection is the most common method. Its basic principle is that each individual's selection probability is positively correlated with its fitness value. Being probability-based and combined with elitist preservation, this method ensures that the current best individual can evolve to the next generation while improving local optima issues.

When evolving from generation τ to $\tau+1$, let π_1^τ be the best individual in the population, with other individuals π_i^τ (where $i \geq 2$) having selection probability:

$$P(\pi_i^\tau) = \frac{E_t(\pi_i^\tau)}{\sum_{j=2}^{p_N} E_t(\pi_j^\tau)}$$

4.2 Crossover Operator

The crossover operator is the primary operator for evolving new individuals in genetic algorithms, significantly impacting search efficiency. To improve convergence speed, this paper employs segmented crossover operators for parallel solving of real-number and binary coding regions. Assuming real-number and binary coding regions perform crossover operations with probabilities p_c^r and p_c^b respectively, and adapting crossover probabilities:

$$p_c^r = \begin{cases} \frac{k_1(E_t^{\max} - E_t')}{E_t^{\max} - \overline{E_t}}, & E_t' \geq \overline{E_t} \\ k_2, & E_t' < \overline{E_t} \end{cases}$$

$$p_c^b = \begin{cases} \frac{k_3(E_t^{\max} - E_t')}{E_t^{\max} - \overline{E_t}}, & E_t' \geq \overline{E_t} \\ k_4, & E_t' < \overline{E_t} \end{cases}$$

where $E_t^{\max}$ is the best individual's fitness, E_t' is the maximum fitness of two chromosomes undergoing crossover, $\overline{E_t}$ is the average fitness of generation τ, and k_1, k_2, k_3, k_4 are constants in $(0, 1]$.

The hybrid coding crossover operator execution process is: a) Randomly pair individuals selected by tournament operator and calculate adaptive crossover probability for the pair. b) Randomly select two crossover points on paired chromosomes, located in real-number and binary coding regions respectively. c) For real-number coding region crossover points, judge whether crossover is needed with probability p_c^r; if yes, exchange genes, otherwise remain unchanged. For binary coding region crossover points, judge with probability p_c^b; if yes, exchange genes, otherwise remain unchanged.

4.3 Mutation Operator

After multiple genetic evolutions, population individuals gradually converge. The mutation operator enhances population diversity and alleviates local optima. Since chromosomes contain two coding regions, this paper proposes a segmented mutation operator. Assuming real-number and binary coding regions mutate with probabilities p_m^r and p_m^b respectively, and adapting mutation probabilities:

$$p_m^r = \begin{cases} \frac{k_1'(E_t^{\max} - E_t)}{E_t^{\max} - \overline{E_t}}, & E_t \geq \overline{E_t} \\ k_2', & E_t < \overline{E_t} \end{cases}$$

$$p_m^b = \begin{cases} \frac{k_3'(E_t^{\max} - E_t)}{E_t^{\max} - \overline{E_t}}, & E_t \geq \overline{E_t} \\ k_4', & E_t < \overline{E_t} \end{cases}$$

where E_t is the fitness of the chromosome undergoing mutation, and k_1', k_2', k_3', k_4' are constants in $(0, 1]$.

The hybrid coding mutation operator execution process is: a) Calculate adaptive mutation probability for the mutation chromosome, randomly select one gene from real-number and binary coding regions respectively. b) With probability p_m^r, judge whether real-number coding region gene needs mutation; if yes, replace with a random number in $[0, 1]$, otherwise no operation. c) With probability p_m^b, judge whether binary coding region gene needs mutation; if yes, replace with 0 or 1, otherwise no operation.

The hybrid coding genetic algorithm for solving equation (10) requires parameters including F_t (available computing resource set), A (DT construction pa-

rameter set), and G (network information set). The algorithm flow is shown in Algorithm 1.

Algorithm 1: Hybrid Coding Genetic Algorithm

```
Input: F_t, A, G
Output: Ω_s, Ω_h, Link
a) Randomly initialize population under constraints and sort individuals by fitness
b) while not converged do
c)   while new individuals < p_N do
       /* Crossover to generate new individuals */
d)   end while
e)   for each individual do
       /* Mutation operation */
f)   end for
g)   Sort population by fitness
h)   if best new individual fitness > $\pi$_1^$\tau$ then
       Replace $\pi$_1^$\tau$ with best new individual
i)   else
       Replace worst new individual with $\pi$_1^$\tau$
j)   end if
k) end while
```

5 Simulation and Performance Analysis

To validate VGDT performance, this paper conducts simulations on DT construction delay, completeness, validity, and communication overhead using PyCharm environment, with detailed results as follows.

5.1 Simulation Environment Setup

To evaluate the proposed VGDT construction technology, we select the DT construction algorithm from reference [14] and the comparison algorithm from reference [24] as baselines. The simulation topology is the Claranet data center network [25], consisting of 15 nodes and 17 edges. As shown in [Figure 3: see original paper], there are 15 switches and 5 edge servers, with the number of hosts under each switch following a uniform distribution of [5, 10]. Data transmission rate is 100Mb/s, with queuing and processing delays on switches both being 5ms. For DT application requirements, switches map 5 types of elements with data volumes (in bytes) uniformly distributed in [0.01M, 0.05M], accounting for a total weight of 0.6 in DT model completeness. Hosts map 10 types of elements with data volumes uniformly distributed in [0.01M, 0.05M], accounting for a total weight of 0.4 in DT model completeness. The conversion ratio between data volume and computing cycles for complete mapping is 100.

5.2 Simulation Results and Performance Analysis

To evaluate VGDT application performance, this paper compares model construction delay, completeness, validity, and communication overhead of VGDT against the SAMPLING algorithm from reference [14] and traditional complete mapping algorithm under different network average available computing resource conditions.

[Figure 4: see original paper] shows the impact of different network computing resources and impact factor α on DT construction communication overhead. When computing resources are scarce, VGDT incurs higher communication overhead, which increases for all algorithms as computing resources grow. When α is small, model construction delay has less impact on model effectiveness, so communication overhead at $\alpha = 5$ is greater than at $\alpha = 1$. To fully utilize network computing resources, VGDT establishes twins for some hosts and switches on edge servers with longer communication distances but more computing resources, causing VGDT communication overhead at $\alpha = 1$ to first increase then decrease when average computing resources reach approximately 10GHz. Moreover, when computing resources are sufficient, VGDT achieves lower communication overhead than comparison algorithms by placing twins on more appropriate edge servers.

[Figure 5: see original paper] shows the impact of different network available computing resources and impact factor α on DT model construction delay. When average computing resources on edge servers are scarce, both VGDT and comparison algorithms exhibit relatively high model construction delay. As network computing resources increase, VGDT improves mapping granularity to enhance model effectiveness, maintaining higher model construction delay. When $\alpha = 5$, the impact of construction delay on effectiveness is greater, resulting in lower delay than at $\alpha = 1$. When $\alpha = 1$, the curves show higher construction delay. As computing resources become more abundant, all three approaches gradually reduce model construction delay.

[Figure 6: see original paper] shows the impact of different network computing resources and impact factor α on DT model completeness. When computing resources are scarce, VGDT achieves higher completeness, but as resources increase, comparison algorithms' completeness rises rapidly, approaching VGDT levels at $\alpha = 1$. Additionally, due to the impact of model construction delay on effectiveness, VGDT's completeness increases more slowly than comparison algorithms as network computing resources grow, though all three algorithms eventually reach similar completeness levels.

[Figure 7: see original paper] shows the impact of different network computing resources and impact factor α on DT model validity. When computing resources are scarce, VGDT demonstrates higher model validity compared to baseline algorithms. However, as computing resources increase, comparison algorithms' validity rises rapidly, eventually approaching VGDT curves. Meanwhile, due to α's influence, all three algorithms achieve higher model validity at $\alpha = 5$ than

at $\alpha = 1$.

Simulation results demonstrate that under different network computing resource conditions, VGDT achieves higher model validity by optimizing mapping granularity and twin placement to fully utilize available network computing resources, thereby constructing more effective DT to better meet application demands.

6 Conclusion

As SDN application scenarios and requirements diversify, digital twin technology can help SDN more accurately analyze, predict, and control networks to better satisfy application demands. Under limited SDN computing resources, this paper proposes a variable granularity digital twin construction technology that adjusts mapping data granularity and twin placement layout based on application scenario demand characteristics. By analyzing the impact of mapping granularity and twin placement on DT model completeness and delay, we establish a variable granularity digital twin optimization model and solve for optimal mapping granularity and twin placement under different computing resources using a hybrid coding genetic algorithm. Simulation results show that according to application demand characteristics, VGDT can fully utilize available network computing resources to construct more effective DT for SDN, thereby improving SDN's ability to adapt to dynamic demands across various application scenarios. Future work will apply VGDT to industrial Internet, IoT, next-generation mobile communication networks, unmanned swarms, and other network environments to develop more targeted, deeply coupled optimization models and algorithms for specific application requirements.

References

[1] Piedrahita A F M, Gaur V, Giraldo J, et al. Leveraging software-defined networking for incident response in industrial control systems [J]. IEEE Software, 2017, 35(1): 44-50.

[2] Al-Rubaye S, Kadhum E, Ni Q, et al. Industrial internet of things driven by SDN platform for smart grid resiliency [J]. IEEE Internet of Things Journal, 2017, 6(1): 267-277.

[3] Manickavasagam B, Amutha B, Priyanka S. Optimal packet routing for wireless body area network using software defined network to handle medical emergency [J]. International Journal of Electrical and Computer Engineering, 2020, 10(1): 427.

[4] Li Kexin, Wang Xingwei, Yi Bo, et al. Survey of intelligent software defined networking [J]. Journal of Software, 2021, 32(1): 118-136.

[5] Tao Fei, Zhang He, Liu Ang, et al. Digital twin in industry: State-of-the-art [J]. IEEE Trans on Industrial Informatics, 2018, 15(4): 2405-2415.

[6] Zhao Liang, Han Guangjie, Li Zhuhui, et al. Intelligent digital twin-based software-defined vehicular networks [J]. IEEE Network, 2020, 34(5): 178-184.

[7] Krishnan P, Jain K, Buyya R, et al. MUD-based Behavioral Profiling Security Framework for Software-defined IoT Networks [J/OL]. IEEE Internet of Things Journal, 2021. (2021-9-17) [2022-01-26]. http://doi.org/10.1109/JIOT.2021.3113577.

[8] Wu Yiwen, Ke Zhang, Yan Zhang. Digital Twin Networks: a Survey [J]. IEEE Internet of Things Journal, 2021, 8(18): 13789-13804.

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

[10] Khan L U, Saad W, Niyato D, et al. Digital-Twin-Enabled 6G: Vision, architectural trends, and future directions [J]. IEEE Communications Magazine, 2022, 60(1): 74-80.

[11] Xu Xiaolong, Shen Bowen, Ding Shengchao, et al. Service offloading with deep Q-network for digital twinning empowered Internet of Vehicles in edge computing [J]. IEEE Trans on Industrial Informatics, 2020, 18(2): 1414-1423.

[12] Zhang Ke, Cao Jiayu, Zhang Yan. Adaptive Digital Twin and Multi-agent Deep Reinforcement Learning for Vehicular Edge Computing and Networks [J]. IEEE Trans on Industrial Informatics, 2021, 18(2): 1405-1413.

[13] Lopez J, Rubio J E, Alcaraz C. Digital Twins for Intelligent Authorization in the B5G-Enabled Smart Grid [J]. IEEE Wireless Communications, 2021, 28(2): 48-55.

[14] Yang Hongwei, Li Yang, Yao Kehan, et al. A Systematic Network Traffic Emulation Framework for Digital Twin Network [C]// IEEE the 1st International Conference on Digital Twins and Parallel Intelligence (DTPI). Beijing: IEEE Press, 2021: 94-97.

[15] Dong Shi. Survey on software defined network security [J]. Computer Science, 2021, 48(3): 295-306.

[16] Tao Fei, Cheng Jiangfeng, Qi Qinglin, et al. Digital twin-driven product design, manufacturing and service with big data [J]. The International Journal of Advanced Manufacturing Technology, 2018, 94(9-12): 3563-3576.

[17] Rathore M M, Shah S A, Shukla D, et al. The Role of AI, Machine Learning, and Big Data in Digital Twinning: A Systematic Literature Review, Challenges, and Opportunities [J]. IEEE Access, 2021, 9: 22041-22060.

[18] Nguyen H X, Trestian R, To D, et al. Digital twin for 5G and beyond [J]. IEEE Communications Magazine, 2021, 59(2): 10-15.

[19] Dong Shi, Sarem M. DDoS attack detection method based on improved KNN with the degree of DDoS attack in software-defined networks [J]. IEEE Access, 2019, 8: 5039-5048.

[20] Hou Xiangwang, Ren Zhiyuan, Wang Jingjing, et al. Reliable computation offloading for edge-computing-enabled software-defined IoV [J]. IEEE Internet of Things Journal, 2020, 7(8): 7097-7111.

[21] Wu Yiwen, Zhou Sipei, Wei Yunkai, et al. Deep reinforcement learning for controller placement in software defined network [C]// IEEE INFOCOM-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). Toronto: IEEE Press, 2020: 1254-1259.

[22] Liu Junmei, Gao Yuelin. Improved differential evolution algorithm for nonlinear mixed integer programming problems [J]. Chinese Journal of Engineering Mathematics, 2010, 27(6): 967-974.

[23] Ma Yongjie, Yun Wenxia. Research progress of genetic algorithm [J]. Application Research of Computers, 2012, 29(4): 1201-1206, 1210.

[24] Lu Yunlong, Maharjan S, Zhang Yan. Adaptive edge association for wireless digital twin networks in 6g [J]. IEEE Internet of Things Journal, 2021, 8(22): 16219-16230.

[25] Chen Junyan, Li Yue, Liang Chuxin, et al. SDN Multi-controller deployment and traffic load balancing [J]. Computer Science & Technology, 2021, 43(5): 830-835.

Note: Figure translations are in progress. See original paper for figures.