This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

IEEE Open Journal of the Communications Society

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/xxxxx

Lagrangian Relaxation Based Parallelized Quantum Annealing and its Application in Network Function Virtualization

WENLU XUAN[1,2] *(Student Member, IEEE)*, ZHONGQI ZHAO [1,2] *(Student Member, IEEE)*, LEI FAN [1,2] *(Senior Member, IEEE)*, AND ZHU HAN[1,3] *(Fellow, IEEE)*

[1]Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204 USA
[2]Department of Engineering Technology, University of Houston, Houston, TX 77204 USA
[3]Department of Computer Science and Engineering, Kyung Hee University, Seoul, 446-701, South Korea

CORRESPONDING AUTHOR: Wenlu Xuan (e-mail: wxuan@uh.edu).

This work was partially supported by NSF CNS-2107216, CNS-2128368, CMMI-2222810, ECCS-2302469, ECCS-2045978, US Department of Transportation, Toyota, Amazon, and Japan Science and Technology Agency (JST) Adopting Sustainable Partnerships for Innovative Research Ecosystem (ASPIRE) JPMJAP2326.

ABSTRACT Quantum computing is commonly considered one highly efficient computing method with the potential to revolutionize computation technology and solve problems that are currently unsolvable. However, due to the limitation of hardware equipment and an immature experimental base, quantum technology is still in its early stages and is far from achieving the expected performance, especially in solving large-scale complex problems. To break through these barriers, we propose a parallelized quantum annealing algorithm based on Lagrangian relaxation. The proposed algorithm divides the large-scale problem into several small problems and then employs multiple quantum computers to solve them. Our proposed approach overcomes the limited number of qubits and allows quantum computing to solve large-scale optimization problems. Additionally, we incorporate a local search method to ensure this Lagrangian relaxation-quantum algorithm achieves an optimal solution. We use the proposed parallelized quantum annealing algorithm to solve optimal scheduling problems in network function virtualization networks. The problem is expressed in a nonlinear optimization model that is NP-hard. Our proposed algorithm presents excellent time performance in solving this virtualized network functions scheduling problem, compared with the Lagrangian relaxation-based classical algorithm.

INDEX TERMS Lagrangian relaxation, quantum algorithm, network function virtualization, quantum computing, virtualized network functions scheduling problem, optimization problem, tabu search algorithm

I. INTRODUCTION

IN recent decades, quantum computing entered people's vision and has made significant strides in its development. In the 1980s, the idea of quantum computing was initially suggested by physicist Richard Feynman [1]. He conceived of building hardware based on quantum theories to simulate quantum dynamic systems. Now researchers are realizing this vision and extending the application of quantum computing to the fields of medical development [2], encryption decoding, communication networks [3], financial planning [4], [5], and also artificial intelligence [6], [7]. Quantum computing follows the laws of quantum mechanics to handle problems so that unique quantum characteristics make quantum computing deliver enhanced computational efficiency. In studying quantum computing technology in depth, researchers find that quantum annealing shows prominent potential superiority in solving optimization problems because the optimal problems have subtle similarities to the fundamental theory of quantum annealing from the mathematical aspect. Based on these advantages, quantum

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

annealing is placed in great hopes to break the bottleneck of classical computing and solve complex problems.

Quantum annealing is predominantly grounded in the Ising model, a mathematical representation of a physical system consisting of interacting spins. During the quantum annealing process, the system evolves according to the laws of quantum mechanics, exploring different spin configurations in parallel. The system eventually reaches its ground state, which represents the optimal solution to the problem at hand. In previous works [8]–[11], quantum annealing has been proven to be effective in solving certain NP-hard problems. Based on these research results and also theoretical analysis, quantum annealing is considered to have the potential to outperform classical algorithms. However, the applications of quantum annealing are restricted due to the limitation of the hardware. To fully utilize the advantage of quantum annealing, some quantum algorithms are designed to assist quantum annealing to tackle optimization problems. Alanis *et al*. [11] proposed a multi-objective dynamic programming framework to solve the Pareto-optimal routing problem in communication networks. The Evolutionary Quantum Pareto Optimization (EQPO) algorithm, which takes advantage of quantum parallelism, was developed based on this dynamic framework. It has been proved that EQPO can find Pareto-optimal routes in polynomial time. Feng *et al*. [12] tried to solve unit commitment problems by a hybrid algorithm based on a quantum approximate optimization algorithm and surrogate Lagrangian Relaxation. This algorithm does not need the optimal dual value to assist the convergence and so it expands the application scope of the algorithm. The Benders' decomposition method was first combined with quantum computing in [13]. Zhao *et al*. leveraged quantum annealers to solve the master problem in Benders' decomposition. In [14], they developed a hybrid quantum-classical multi-cuts Benders' decomposition (HQCMBD) algorithm based on the previous paper and applied it to solving the mixed integer linear programming (MILP) model of the data center energy management problem. The simulation results show that the HQCMBD algorithm has better performance in aspects of success rate, robustness, and so on. Spirited by these works, and also to mitigate the issue of limited qubits, we proposed a parallelized quantum annealing algorithm that integrates quantum annealing with the Lagrangian relaxation technique to reduce the need for qubits so that quantum annealing can be employed to solve complex optimization problems. Additionally, this algorithm incorporates a tabu search method to assist the search for the global optimum. In this paper, to validate the practicality and effectiveness of our proposed algorithm, we use the proposed algorithm to solve optimization problems in network function virtualization (NFV) networks.

In recent years, NFV technology, as a transformative technology of network management, has been attached to more importance and also more widely used in real-world scenarios [15], [16]. NFV offers a virtualized environment that allows network functions to operate on software platforms, thereby enhancing the efficiency of network resource utilization [17]. In NFV network, virtualized network functions (VNFs), virtualizing network services, are typically deployed on one or more virtual machines (VMs) that run on standard servers or in clouds [18], [19]. A crucial challenge in the networks is to efficiently schedule VNFs in response to users' requests, which is generalized as the VNFs scheduling problem. The VNFs scheduling problem is intricate and computationally demanding due to the numerous variables and constraints. In previous work [3], we built an integer linear programming (ILP) model to address a simplified VNFs scheduling problem and leveraged quantum annealers to solve it. Subject to the limited qubits and other hardware constraints, quantum annealers cannot solve this problem in large-scale cases and the simulation results presented instability as the number of variables increased. To address this issue, we develop a parallelized quantum annealing algorithm in this paper. We present a more complex system model of the NFV network and build an optimization model of the VNFs scheduling problem based on it. This ILP model has the objective function that minimizes the total delay of all service chains, which includes processing delay on VMs and transmission delay between them. The simulation experiments demonstrate that our algorithm can efficiently solve the VNFs scheduling problem on a large scale. The contributions of this paper are summarized as follows:

1) We propose a novel parallelized quantum annealing algorithm, which employs Lagrangian relaxation to decompose problem models and leverages quantum annealing to solve the sub-problem models. It also includes the tabu search method that is used to assist in converging to the optimal solution of the original model.
2) An ILP model of the VNFs scheduling problem is constructed. This optimization model only with binary variables has the object of minimizing the total processing delay and transmission delay in the NFV network.
3) We conduct simulation experiments to investigate the performance of the proposed parallelized quantum annealing algorithm and obtain excellent results that indicate its superiority in solving efficiency and robustness compared to the Lagrangian relaxation-based classical algorithm.

The rest of this paper is organized as follows. Section II describes our proposed algorithm and elucidates its crucial parts in detail. Section III introduces the system model and ILP formulations of the VNFs scheduling problem. This section also presents steps for decomposing this optimal problem into two sub-problems using Lagrangian relaxation, followed by the corresponding transformed QUBO models. Section IV showcases the simulation results and their analysis. Finally, Section V concludes the paper.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

II. PROPOSED LAGRANGIAN RELAXATION BASED PARALLELIZED QUANTUM ANNEALING ALGORITHM

Existing quantum computers cannot be extensively used as expected because their applications are currently hindered by several significant limitations, including the limited number of available qubits, the stability of quantum systems, and embedding techniques. To overcome these difficulties and fully exploit quantum annealing for large-scale problem-solving, we develop a novel parallelized quantum annealing algorithm combining quantum annealing, Lagrangian relaxation, and the tabu search method. In this section, we introduce these three crucial techniques and explain the proposed algorithm in detail. In Subsection A, there is a brief introduction to quantum annealing principles and a clear analysis of the annealing process. Subsection B shows how to use the Lagrangian relaxation method to decompose the programming model. Subsection C briefly explains the tabu search algorithm. In Subsection D, our proposed algorithm is developed on these computing techniques, and its framework is shown.

A. QUANTUM ANNEALING

Quantum annealing is similar to the classical algorithm, simulated annealing. Simulated annealing is a famous heuristic algorithm designed to solve optimization problems by searching the solution space through thermal fluctuations. In theory, quantum annealing should perform better than simulated annealing because quantum annealing leverages quantum fluctuations rather than thermal fluctuations to get the optimal solution [20]. Quantum annealing applies quantum efforts like quantum tunneling to accelerate searching the ground state of an Ising Hamiltonian which corresponds to the optimal solution of the optimization problems. This process follows the quantum evolution of the time-dependent Schrödinger equation. The Schrödinger equation is

$$i\frac{d}{dt}|\Psi(t)\rangle = H(t)|\Psi(t)\rangle, \tag{1}$$

where $\psi(t)$ is the state vector of the quantum system and the $H(t)$ is the Hamiltonian operator. This equation is the general description of the quantum system evolution. It is a great challenge to solve this equation directly because the computational complexity increases exponentially when the problem complexity increases. Quantum annealing basically follows adiabatic evolution, which points out that the quantum system will stay in the ground state if the Hamiltonian related to the dynamics changes slowly enough.

To obviously analyze the quantum annealing process, the Hamiltonian of the whole quantum system in adiabatic evolution is briefly explained as

$$H(t) = A(t)H_i + B(t)H_f, \tag{2}$$

where $A(t)$ and $B(t)$ are time-dependent prefactors and they control the evolution process [21], [22]. H_i and H_f are the initial Hamiltonian and final Hamiltonian, respectively. Both can be expressed by linear combinations of several eigenstates. At the beginning of the adiabatic evolution, the whole system is at the minimum energy eigenstate of the initial Hamiltonian state. In other words, $A(0) = 1$ and $B(0) = 0$. In the evolution process, $A(t)$ gradually decreases while $B(t)$ gradually increases so $A(t)$ and $B(t)$ are supposed to be differentiable. Finally, at time T, the end of the annealing process, $A(T)$ reaches 0, and $B(T)$ arrives at 1, which means the whole quantum system finishes the state change from the initial Hamiltonian state to the ground state of the final Hamiltonian. The ground state is the eigenstate with the lowest eigenvalue, which means the whole system is at the lowest energy eigenstate. Finally, the system will stay at the ground state of the final Hamiltonian in the ideal case.

For quantum computing, the final Hamiltonian can be represented by the Ising model with operators σ_i^z, which shows the interaction state of n qubits in a 2^n dimension Hilbert space. If qubits are prepared as spins, σ_i^z is the spin component state of spins projected on axis z component. The Ising model can be expressed as

$$H = \sum_i h_i\sigma_i^z + \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z, \tag{3}$$

where σ_i^z can be set as 1 or -1. h_i indicates how the external field affects the qubit σ_i^z. J_{ij} is the coupling parameter of σ_i^z and σ_j^z. The external field can have effects on the spin state which can be spin-up and spin-down. The two spin states have different energy levels. The sign of h_i represents which spin state is more preferred under the influence of the external field. The value of h_i is determined by the field strength and the energy of the spin particle. Every spin can be considered as a magnetic dipole that can produce a small magnetic field so every spin is influenced by its neighbor magnetic field. The sign of J_{ij} shows whether these spins prefer to be aligned or anti-aligned with their neighbors. It determines the ferromagnetic characteristics of the substance. The value of J_{ij} indicates the coupling strength of the neighbor spins. It mostly depends on the distance between these two interactional spins.

For the whole process of quantum annealing, the Ising model can be expressed as

$$\begin{aligned} H(s) = &-\frac{A(s)}{2}\left(\sum_i \sigma_i^x\right) \\ &+ \frac{B(s)}{2}\left(\sum_i h_i\sigma_i^z + \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z\right), \end{aligned} \tag{4}$$

where σ_i^x is the spin component state of spins projected on axis x component. $A(s)$ and $B(s)$ are two prefactors that change along time. $s = t/t_{\text{annealing}}$, which means s denotes the progress of the whole annealing process. $t_{\text{annealing}}$ is the total time of quantum annealing. The first term is the initial Hamiltonian which is generally set at the lowest-energy eigenstate in practice. The second term is the final Hamiltonian. Before starting quantum annealing, the system is situated in the lowest-energy eigenstate of the initial

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

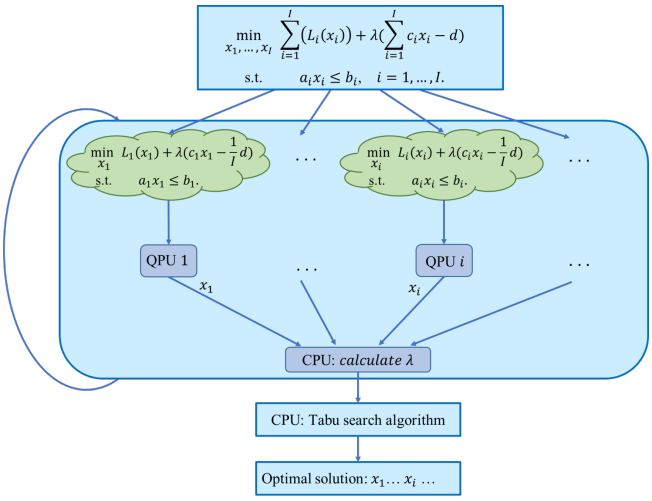


FIGURE 1. The framework of the proposed Lagrangian relaxation based parallelized quantum annealing algorithm.

Hamiltonian. In the quantum annealing process, the state of spins is changed and their interaction makes efforts on the Hamiltonian of the whole quantum system. Consequently, the final Hamiltonian is produced and then the whole system tends to evolve into the lowest-energy eigenstate of the final Hamiltonian, which is also called the ground state of the final Hamiltonian. In ideal conditions, the evolution process adheres to the adiabatic evolution, which presents that the whole system always stays at the lowest-energy state so it is easy to achieve the lowest-energy eigenstate of the final Hamiltonian. However, the quantum annealing process is fraught with complexities, and there may be other energy levels close to the ground state. The system may decide to move to these energy levels and stay at these energy levels at the end of the quantum annealing. Consequently, the system may only find the local minimum of the objective function rather than the global minimum at the end.

B. LAGRANGIAN RELAXATION

The Lagrangian relaxation technique is a crucial tool in assisting to solve optimization problems [23]–[25]. It works based on the separability of problem models and the decomposition makes the models much easier to solve due to removing the complex constraints. The method involves relaxing the coupling constraints through the use of Lagrange multipliers, which leads the models to approach the optimal solutions of original problems in the solving process.

Assume there is an integer programming model shown as

$$L_{\text{IP}} = \min_{x_1,\ldots,x_I} \quad \sum_{i=1}^{I} L_i(x_i) \tag{5a}$$

$$s.t. \quad a_i x_i \le b_i, \qquad i = 1,\ldots,I, \tag{5b}$$

$$\sum_{i=1}^{I} c_i x_i \le d, \tag{5c}$$

where x_i are variables, and a_i, b_i, c_i and d are constants. The objective function can be divided into I terms based on the variables x_i. There are I constraints, which make the IP model hard to solve. Thus, Lagrangian relaxation is applied to relax these constraints and then add them to the objective function. The new model is

$$L_{\text{LR}}(\lambda) = \min_{x_1,\ldots,x_I} \quad \sum_{i=1}^{I} \left(L_i(x_i) + \lambda(\sum_{i=1}^{I} c_i x_i - d) \right) \tag{6a}$$

$$s.t. \quad a_i x_i \le b_i, \qquad i = 1,\ldots,I. \tag{6b}$$

This new model can be divided into I sub-models that everyone only has variables x_i and then the surrogate subgradient method is leveraged to update λ. The process is briefly shown in Fig. 1. In our proposed algorithm, this process is structured into two distinct levels, the low level and the high level. At the low level, the sub-models are individually solved by quantum computing and get the optimal solution and also the value of x_i, which is used to update λ. At the high level, the coordination between the sub-problems is performed through the updating of the Lagrange multiplier λ. The iterations of updating λ and solving sub-models cannot end until the solutions tend to converge. At the same time, the optimal feasible solution of the original model is achieved. The Lagrangian relaxation method is easier to use in many cases and the process is convenient to adjust depending on problems.

C. TABU SEARCH ALGORITHM

The tabu search algorithm is a neighborhood search algorithm and it is widely used in finding the approximate solution to NP-hard combinatorial optimization problems [26]–[28]. The tabu search algorithm continuously searches for better solutions in the neighborhood by iterations and then replaces the current solution to achieve optimal solutions step by step. During the search process, the information of found solutions is recorded in the tabu list. This tabu list not only prevents the algorithm from cycling back to already explored solutions but also enables it to escape local optima by allowing moves that may seem non-beneficial. Thus, this mechanism significantly enhances the efficiency and effectiveness of the search process. The tabu search algorithm has a strong dependence on the initial solution, and a good initial solution can markedly shorten the search process. In our proposed algorithm, the final result achieved by the Lagrangian relaxation based parallelized quantum annealing algorithm is a good initial solution. Therefore, we choose this method to rapidly converge the solution found by quantum annealing to the global optimum with reasonable parameter settings.

D. PROPOSED ALGORITHM

We propose a Lagrangian relaxation based parallelized quantum annealing algorithm based on Lagrangian relaxations to solve large-scale optimization problems. The workflow protocol of the algorithm is shown in Fig. 2. We first

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

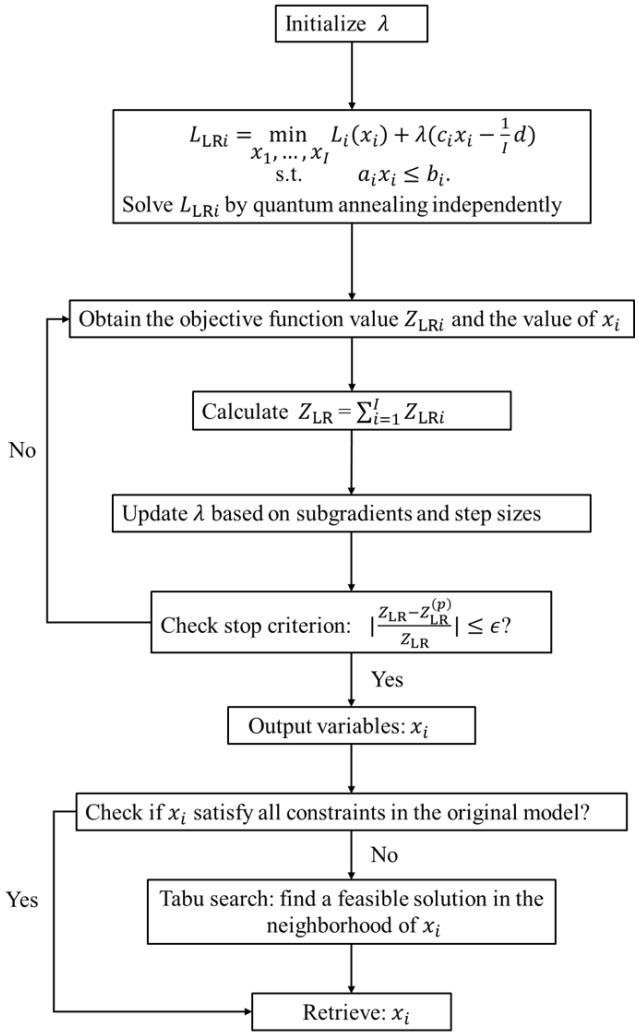


FIGURE 2. **The flowchart of the proposed Lagrangian relaxation based parallelized quantum annealing algorithm.**

build the integer programming model and then decompose it by Lagrangian relaxations to form several sub-problem models $L_{\mathrm{LR}i}$. Then we use quantum annealing to solve these sub-problem models $L_{\mathrm{LR}i}$ separately and get the optimal solutions to these models and the value of x_i at these points. The value of λ is updated based on the subgradient methods. Finally, we check if the objective function value Z_{LR} can satisfy the stop criterion or not. $Z_{\mathrm{LR}}^{(p)}$ denotes the objective function value got in the previous iteration. ϵ is a sufficiently small value. Thus, the solving process stops repeating when the solution result of every iteration does not change. If the stop criterion cannot be reached, the updated λ will replace the previous one and then we will solve sub-problem models again. Actually, quantum annealing may lead to the issue of achieving the local optimal solution rather than the global optima because of the limit of hardware equipment. Furthermore, the Lagrangian relaxation method only reaches the approximate optimal solution of the original problem. Thus, we design post-process steps that leverage tabu search

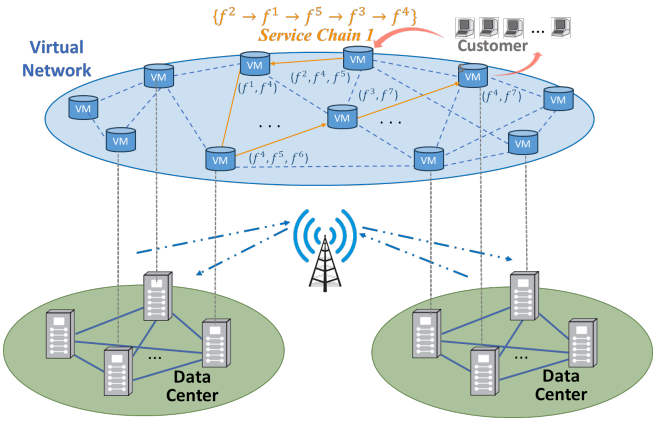


FIGURE 3. **The diagram of the NFV network.**

after quantum annealing in our algorithm to help converge and achieve the optimal solution. In Section V, we use the proposed Lagrangian relaxation based parallelized quantum annealing algorithm to solve an optimization problem in communication networks and explain this process in detail.

III. USE CASE IN COMMUNICATION NETWORK

NFV emerges to standardize functions in a wireless network and enhances the flexibility and scalability of networks to accommodate new services efficiently. Among the various optimization problems inherent in NFV networks, network resource allocation problems, especially the VNFs scheduling problem, stand out. The VNFs scheduling problem is arranging VMs to process functions of service chains to minimize the delay of all service chains' processing. In Subsection A, we explain the system model of NFV networks and build the integer linear programming (ILP) model to describe the VNFs scheduling problem. In Subsection B, the Lagrangian relaxation method is employed to divide the whole problem into two subproblems. These subproblems are transformed into QUBO models in Subsection C. Finally, we operate the proposed algorithm mentioned in Section II to solve the VNFs scheduling problem. The whole solving process is concisely described in Subsection D.

A. SYSTEM MODEL AND ILP FORMULATION

In the NFV network system, as shown in Fig. 3, the hardware providing computing and storage resources usually is abstracted as VMs and these VMs connect through virtual links. Each VM can run several virtual functions to conduct computing following users' requests. If a user request requires processing large amounts of data among different data centers, the processing delay at the VMs and the transmission delay between them cannot be ignored. A user request is fulfilled through the operation of a service chain, which combines several VNFs in the network. Hence, the total delay in processing the request is the time it takes for data to be processed by VNFs of the service chain and the transmission delay between the corresponding VMs. The model proposed in this paper is built to provide the best

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

arrangement to minimize the total delay of all activated service chains in the network.

We abstract the system model from the real network system. It is assumed that there are several types of VMs in the network and VMs can provide some different virtual functions f^k. These functions f^k, with k indicating the type of functions, may be operated on more than one type of VMs. For example, VM 1 can process function f^1, f^3 and f^4. f^3 can be served on VM 1, VM 2 and VM 5. In the model, the number of VMs in the network is M, and also m and n are indexes of VMs. We divide the running time of VMs, denoted as $T_{\max}$, into multiple time slots, and every time slot has the length of ΔT. Every VM can only process one function in one time slot. In the NFV network, VMs communicate with each other by one virtual link. The virtual link between VM m and VM n is denoted as $l_{(m,n)}$. Every virtual link can only serve data transmitting for one instance of the virtual function at the same time. When the NFV network receives the users' request, the service chains will be invoked to process users' data. As shown in Fig. 3, the service chain 1 is used to meet the request of users. The virtual functions of service chain 1 are operated on VMs that can provide corresponding VNFs.

There are I service chains that are ready to process data following the customers' requests submitted to the NFV network. Service chains can be regarded as sequences of virtual functions with specific orders. In our system model, all virtual functions and service chains are instantized to clearly denote different individuals. These instances of virtual functions are denoted as f_{ij}^k, which means the virtual function is the j_{th} in the service chain i and it is divided into the k_{th} type of function. All VMs that can process f_{ij}^k compose the set V_{ij}^k. Appropriate VM is chosen from this set to process these f_{ij}^k. In the system model, the processing time of f_{ij}^k is denoted as t_{ijm}, which is calculated by W_{ijm}/C_m. The size of data that needs to be processed is W_{ijm} and the computing rate of VM m is denoted as C_m. T_{ijm} is the number of time slots, which the total time length corresponds to t_{ijm}. $T_{ij(m,n)}$ is the number of time intervals occupied by the transmission of f_{ij}^k processing results through the virtual link $l_{(m,n)}$. Based on the system model, we build an ILP model with binary decision variables, x_{ijm}, y_{ijmt}, z_{ijmt}, p_{ijmt}, u_{ijmn}, and v_{ijmnt}, to minimize the total delay including processing delay and transmission delay in processing customers' requests. All properties of the system model are expressed in the constraints of the ILP model. Constraints related to arranging VMs to employ virtual functions to process data are listed in Appendix A equation (48)-(58), which are also explained in [3]. Decision variables, x_{ijm}, y_{ijmt}, z_{ijmt}, and p_{ijmt}, are introduced to formulate these constraints. One other constraint is shown as

$$u_{ij(m,n)} \leq x_{ijm}, \quad \forall i, j, m, n. \tag{7}$$

If x_{ijm} equals 1, it signifies that VM m is used to process the virtual function f_{ij}^k. If $u_{ij(m,n)}$ equals 1, it means that the virtual link $l_{(m,n)}$ is used to transmit the results of f_{ij}^k. Thus, constraint (7) shows that the link $l_{(m,n)}$ may be chosen to transmit the result of j^{th} function in service i because the VM m is chosen to process the function f_{ij}^k. Constraint (7) indicates the relationship between x_{ijm} and $u_{ij(m,n)}$. If VM m conducts processing function f_{ij}^k, which means $x_{ijm} = 1$, then the link $l_{(m,n)}$ can be used to transmit the result, which denotes as $u_{ij(m,n)} = 1$. There is one other constraint shown as

$$\sum_{m=1}^{N} u_{ij(m,n)} = \sum_{m'=1}^{N} u_{i(j+1)(n,m')}, \quad \forall i, j; \quad n \in V_{i(j+1)}^{k'}. \tag{8}$$

Constraint (8) shows that the VM n must be the end point of the link that is used to transmit the result of j^{th} function in service i and also it must be the start point of the link that is responsible to transmit the result of $(J+1)^{th}$ function. In other words, constraint (8) keeps the continuity of the service chain. There is also one constraint shown as

$$\sum_{m \in V_{ij}^k} \sum_{n \in V_{i(j+1)}^{k'}} u_{ij(m,n)} = 1, \quad \forall i, j. \tag{9}$$

Constraint (9) limits that there is only one link that can be occupied to transmit the result of any function f_{ij}^k. Thus, there should be only one $u_{ij(m,n)}$ that equals to 1 for all possible values of m and n. One other constraint is shown as

$$(1 - u_{ij(m,n)}) \cdot v_{ij(m,n)t} = 0, \quad \forall i, j, m, n, t. \tag{10}$$

If $v_{ij(m,n)t}$ equals 1, it means that the virtual link $l_{(m,n)}$ starts to transmit the results of f_{ij}^k at the beginning of the time slot t. Constraint (10) indicates that $v_{ij(m,n)t}$ could be 1 if and only if $u_{ij(m,n)}$ equals to 1. It is because only when the link is chosen to transmit the result of any functions, the link will serve this transmission in the time slot t. There is one other constraint shown as

$$\sum_{i=1}^{I} \sum_{j=1}^{J} v_{ij(m,n)t} + \sum_{i'=1}^{I} \sum_{j'=1}^{J} v_{i'j'(n,m)t} \leq 1. \quad \forall m, n, t. \tag{11}$$

Constraint (11) presents that there is only one transmission allowed between VM m and VM n in the time slot t. For example, if the virtual link $l_{(m,n)}$ is used to transmit the processing results of f_{ij}^k in the time slot t, which means $v_{ij(m,n)t} = 1$, both $l_{(m,n)}$ and $l_{(n,m)}$ cannot transmit other data in this time slot. There is also one constraint shown as

$$\sum_{t=1}^{T_{\max}} v_{ij(m,n)t} = T_{ij(m,n)} \cdot u_{ij(m,n)}, \quad \forall i, j; \quad m \in V_{ij}^k, \quad n \in V_{i(j+1)}^{k'}. \tag{12}$$

Constraint (12) indicates that the result of the function f_{ij}^k must be transmitted reaching the required transmission time $T_{ij(m,n)}$. If and only if VM m can process function f_{ij}^k and

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

VM n can process function $f_{i(j+1)}^{k'}$, the link $l_{(m,n)}$ can be selected, and then there should be the presence of $T_{ij(m,n)}$. There is one other constraint shown as

$$\sum_{\alpha=1}^{T_{\max}} p_{ijm(t-\alpha+1)} \geq \sum_{n \in V_{i(j+1)}^{k'}} v_{ij(m,n)t}, \quad \forall i,j,t; \quad m \in V_{ij}^{k}. \tag{13}$$

Constraint (13) ensures that starting the transmission of function results must be after completing processing this function. Thus, there must be one $p_{ijmt'}$ that equals 1 in previous time slots when we find $v_{ij(m,n)t} = 1$. One other constraint is shown as

$$\sum_{\beta=1}^{T_{\max}} z_{i(j+1)n(t+\beta)} \geq \sum_{m \in V_{ij}^{k}} v_{ij(m,n)t}, \quad \forall i,j,t; \quad n \in V_{i(j+1)}^{k'}. \tag{14}$$

If $z_{i(j+1)nt}$ equals 1, it signifies that VM n starts to process the virtual function $f_{i(j+1)}^{k'}$ at the beginning of the time slot t. Constraint (14) guarantees that starting processing $f_{i(j+1)}^{k'}$ must be after receiving the results of last function f_{ij}^{k}. Therefore, if there is one $z_{i(j+1)nt}$ that equals 1, we can find $v_{ij(m,n)t'} = 1$ in previous time slots. One other constraint is shown as

$$u_{ij(m,n)} = v_{ij(m,n)t} = 0, \quad \forall i,j,t; \quad m \notin V_{ij}^{k} \quad or \quad n \notin V_{i(j+1)}^{k'}. \tag{15}$$

Constraint (15) shows that if VM m cannot process function f_{ij}^{k} or VM n cannot process function $f_{i(j+1)}^{k'}$, which implies that neither $l_{(m,n)}$ nor $l_{(n,m)}$ will not be used to transmit the results of function f_{ij}^{k}, $v_{ij(m,n)t}$ and $u_{ij(m,n)}$ must be 0.

These above constraints (7) to (15) describe how the users' data is transmitted along the service chain between different VMs. One other constraint is shown as

$$s_{iJ} = \sum_{m=1}^{M} \sum_{t=1}^{T_{\max}} p_{iJmt} \cdot (t-1) \cdot \Delta T, \quad \forall i. \tag{16}$$

Eq. (16) is used to calculate s_{iJ}, which represents the completion time of processing f_{iJ}^{k} that is the last function of the service chain i. $p_{iJmt} = 1$ means that VM m completes processing f_{iJ}^{k} at the beginning of the time slot t. Therefore, the finishing time of the service chain i can be known by calculating the equation (16). All in all, the ILP model of VNFs scheduling problem can be expressed as

$$Z_{\text{IP}} = \min_{\mathbf{s}} \quad \sum_{i=1}^{I} s_{iJ}, \quad s.t. \quad (7)-(16), (48)-(58), \tag{17}$$

where constraint (48)-(58) are listed in Appendix A. The objective function aims to minimize the total processing time of all service chains. In the following subsection, the Lagrangian relaxation method is leveraged to decompose this ILP model and transform these sub-problem models into QUBO forms.

B. LAGRANGIAN RELAXATION

We combine (16) and (17), and have

$$Z_{\text{IP}} = \min_{\mathbf{p}} \quad \sum_{i=1}^{I} \sum_{m=1}^{M} \sum_{t=1}^{T_{\max}} p_{iJmt} \cdot (t-1) \cdot \Delta T. \tag{18}$$

We use Lagrangian relaxation method to transform (7), (13), and (14), and then add them to the original objective function. Now the objective function should be

$$\begin{aligned} Z_{\text{IP}} = \min_{\mathbf{x},\mathbf{p},\mathbf{z},\mathbf{u},\mathbf{v}} \quad & \sum_{i=1}^{I} \sum_{m=1}^{M} \sum_{t=1}^{T_{\max}} p_{iJmt} \cdot (t-1) \cdot \Delta T \\ & + \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{m \in V_{ij}^{k}} \sum_{t=1}^{T_{\max}} \lambda_{ijmt} \Big(\sum_{n \in V_{i(j+1)}^{k'}} v_{ij(m,n)t} \\ & - \sum_{\alpha=1}^{T_{\max}} p_{ijm(t-\alpha+1)} \Big) \\ & + \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{n \in V_{i(j+1)}^{k'}} \sum_{t=1}^{T_{\max}} \lambda_{ijnt} \Big(\sum_{m \in V_{ij}^{k}} v_{ij(m,n)t} \\ & - \sum_{\beta=1}^{T_{\max}} z_{i(j+1)n(t+\beta)} \Big) \\ & + \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{m=1}^{N} \sum_{n=1}^{N} \lambda_{ijmn} (u_{ij(m,n)} - x_{ijm}), \end{aligned} \tag{19}$$

with

$$\lambda_{ijmn} \geq 0, \quad \forall i,j,m,n, \tag{20}$$

$$\lambda_{ijmt} \geq 0, \quad \forall i,j,t; \quad m \in V_{ij}^{k}, \tag{21}$$

$$\lambda_{ijnt} \geq 0, \quad \forall i,j,t; \quad n \in V_{i(j+1)}^{k'}. \tag{22}$$

Lagrange multipliers λ_{ijmn}, λ_{ijmt}, and λ_{ijnt} are used to control the influence of these three constraints on the new objective function. These constraints are relaxed through the transformation to reduce the complexity of the original ILP model. This new optimization problem can be divided into two sub-problems. The objective function of the first sub-problem is

$$\begin{aligned} Z_{\text{IP1}} = \min_{\mathbf{x},\mathbf{p},\mathbf{z}} \quad & \sum_{i=1}^{I} \sum_{t=1}^{T_{\max}} \Big(\sum_{m=1}^{M} p_{iJmt} \cdot (t-1) \cdot \Delta T \\ & - \sum_{j=1}^{J} \sum_{m \in V_{ij}^{k}} \lambda_{ijmt} \sum_{\alpha=1}^{T_{\max}} p_{ijm(t-\alpha+1)} \\ & - \sum_{j=1}^{J} \sum_{n \in V_{i(j+1)}^{k'}} \lambda_{ijnt} \sum_{\beta=1}^{T_{\max}} z_{i(j+1)n(t+\beta)} \Big) \\ & - \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{m=1}^{N} \sum_{n=1}^{N} \lambda_{ijmn} x_{ijm}. \end{aligned} \tag{23}$$

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

The first sub-problem is with constraints (48)-(58) and (20)-(22). This sub-problem is to give the optimal assignment of VMs to process these service chains following the system model. The objective function of the second sub-problem is

$$\begin{aligned} Z_{\text{IP2}} = \min_{\mathbf{u},\mathbf{v}} \quad & \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T_{\max}} \\ & \Big(\sum_{m\in V_{ij}^{k}} \lambda_{ijmt} \sum_{n\in V_{i(j+1)}^{k'}} v_{ij(m,n)t} \\ & + \sum_{n\in V_{i(j+1)}^{k'}} \lambda_{ijnt} \sum_{m\in V_{ij}^{k}} v_{ij(m,n)t} \Big) \\ & + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{m=1}^{N}\sum_{n=1}^{N} \lambda_{ijmn} u_{ij(m,n)}. \end{aligned} \tag{24}$$

The second sub-problem has constraints (8)-(12), (15), and (20)-(22). This sub-problem aims to arrive at the optimization of total transmission time while neglecting processing delay.

We use the subgradient iterative technique to update the set of Lagrange multipliers λ_{ijmn}, λ_{ijmt}, and λ_{ijnt}. In every iteration k, λ_{ijmn}, λ_{ijmt}, and λ_{ijnt} are updated to help converge to the optimal solution. The equations we used to update λ_{ijmn} are shown as

$$g_1^{(k+1)} = u_{ij(m,n)}^{(k)} - x_{ijm}^{(k)}, \tag{25}$$

$$\lambda_{ijmn}^{(k+1)} = \max\left(0, \lambda_{ijmn}^{(k)} + g_1^{(k)}\gamma_1^{(k)}\right). \tag{26}$$

$g_1^{(k)}$ is the subgradient that along to direction of constraint (7) in the k_{th} iteration. The subgradient $g_1^{(k)}$ and the step size $\gamma_1^{(k)}$ control the change of λ_{ijmn}. We also use the same method to calculate λ_{ijmt} and λ_{ijnt}. These equations are shown as

$$g_2^{(k)} = \sum_{n\in V_{i(j+1)}^{k'}} v_{ij(m,n)t}^{(k)} - \sum_{\alpha=1}^{T_{\max}} p_{ijm(t-\alpha+1)}^{(k)}, \tag{27}$$

$$\lambda_{ijmt}^{(k+1)} = \max\left(0, \lambda_{ijmt}^{(k)} + g_2^{(k)}\gamma_2^{(k)}\right), \tag{28}$$

$$g_3^{(k)} = \sum_{m\in V_{ij}^{k}} v_{ij(m,n)t}^{(k)} - \sum_{\beta=1}^{T_{\max}} z_{i(j+1)n(t+\beta)}^{(k)}, \tag{29}$$

$$\lambda_{ijnt}^{(k+1)} = \max\left(0, \lambda_{ijnt}^{(k)} + g_3^{(k)}\gamma_3^{(k)}\right). \tag{30}$$

(g_2, γ_2) and (g_3, γ_3) are pairs of problem subgradients and step size parameters related to constraints (13) and (14). We initially set Lagrange multipliers and then they updated according to equations (25)-(30) in iterations. After finishing every iteration, we calculate the solution of the whole problem model (17). If the results of any iteration reach the stop criterion, we will terminate the iteration and get the final solutions. Before using the proposed algorithm to solve this problem, we also need to transform these two sub-problem models into QUBO models independently.

C. QUBO MODEL

The QUBO model distinguishes itself by comprising solely an objective function that includes quadratic terms, devoid of explicit constraints. As a result, to convert the ILP model into a QUBO model, it is essential to transfer all constraints into equivalent quadratic penalties which are then incorporated into the objective function. These transformations ensure that the restrictive conditions of the original constraints are preserved within the objective function, thereby guiding the model towards reaching the optimal solution. We transform all constraints following the rules shown in Table 1. x_1, x_2 and x_3 are used to denote binary decision variables of original constraints. r_l denotes the binary slack variable introduced in penalty terms. a_l and b are constants. P is the penalty coefficient, which is a sufficiently large positive constant.

TABLE 1. List of Constraint-Penalty Pairs

Constraint	Equivalent Penalty
$x_1 + x_2 = 1$	$P(x_1 + x_2 - 1)^2$
$x_1 + x_2 + x_3 \leq 1$	$P(x_1x_2 + x_1x_3 + x_2x_3)$
$x_1 + x_2 \leq x_3$	$P(x_1 + x_2 - x_3 + \sum_l a_l r_l)^2$
$x_1 + x_2 = b$	$P(x_1 + x_2 - b)^2$

We transform all constraints to penalty terms and all penalty terms of the QUBO model are listed below. We transform (48) and (49) to equations shown as

$$P_{1ij}\left(\sum_{m\in V_{ij}^{k}} x_{ijm} - 1\right)^2, \quad \forall i, j, \tag{31}$$

$$P_{1ijm}\left(\sum_{t=1}^{T_{\max}} z_{ijmt} - x_{ijm}\right)^2, \quad \forall i, j; \quad m \in V_{ij}^{k}. \tag{32}$$

Eq. (31) is transformed from (48). When and only when one x_{ijm} equals 1 with $m \in V_{ij}^{k}$, this penalty with penalty coefficient P_{1ij} will not add a big positive value to the objective function. This design ensures that, to find the minimum, the QUBO model will not allow more than one x_{ijm} to equal to 1. Thus, (31) has the same function with (48). Eq. (32) is transformed from (49). This penalty requires that x_{ijm} and $\sum_{t=1}^{T_{\max}} z_{ijmt}$ must have the same values so it has the same effect as constraint (49) on the problem models. We transform (50) to the equation shown as

$$P_{mt}\left(\sum_{(i\neq i')\vee(j\neq j')} (y_{ijmt} \cdot y_{i'j'mt})\right), \quad \forall m, t. \tag{33}$$

Eq. (33) is equivalent to (50). Constraint (50) ensures that only one y_{ijmt} may be chosen to be set as 1 in any

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

case. If this constraint does not be followed, the terms in (33) will add a large constant to the objective function. This mechanism will lead the QUBO model to follow constraint (50). We transform (51) to the equation shown as

$$P_{1ijmt}\Big(y_{ijmt}^2 - x_{ijm}y_{ijmt}\Big), \quad \forall i,j,m,t. \tag{34}$$

y_{ijmt} is binary variable so y_{ijmt} is equal to the squre of y_{ijmt}. This characteristic of binary variables facilitates the direct transformation of constraint (51). If the values of x_{ijm} and y_{ijmt} don't obey constraint (51), these terms in (34) will lead the solution away from the minimum. We transform (52) to the equation shown as

$$P_{2ijm}\left(\sum_{t=1}^{T_{\max}} y_{ijmt} - T_{ijm}x_{ijm}\right)^2, \quad \forall i,j; \quad m \in V_{ij}^k. \tag{35}$$

Eq. (35) will add a large value to the objective function with the assistance of a penalty if constraint (52) is not satisfied, which means the right-hand side terms have different values from the left-hand side terms. Thus, the optimizer will try to set values of x_{ijm} and y_{ijmt} to obey the constraint (52). We transform (53) to the equation shown as

$$P_{2ijmt}\Big(z_{ijmt}\cdot p_{ijmt}\Big), \quad \forall i,j,m,t. \tag{36}$$

If both z_{ijmt} and p_{ijmt} are equal to 1, terms in (36) will add a large constant to the objective function, so the optimizer will avoid this case. That is what constraint (53) tries to do. Thus, we can transform (53) into the terms in (36). We transform (54), (55) and (56) to the equations shown as

$$P_{3ijmt}\Big(y_{ijm(t-1)} - y_{ijmt} + z_{ijmt} - p_{ijmt}\Big)^2, \quad \forall i,j,m,t. \tag{37}$$

$$P_{4ijmt}\left(\sum_{\alpha=1}^{T_{ijm}} z_{ijm(t-\alpha+1)} - y_{ijmt} + r_{1ijmt}\right)^2, \quad \forall i,j,t; \quad m \in V_{ij}^k. \tag{38}$$

$$P_{1ijm't}\left(z_{i(j+1)m't} - \sum_{m\in V_{ij}^k}\sum_{\beta=1}^{T_{\max}} p_{ijm(t-\beta+1)} + r_{ijm't}\right)^2, \quad \forall i,j,t; \quad m' \in V_{i(j+1)}^{k'}. \tag{39}$$

Eq. (37) can force the solution to follow the constraint (54). Eq. (38) is transformed from (55) by adding a binary slack variable r_{1ijmt}. Eq. (39) is equivalent to (56) and slack variables are also needed in this transformation. We only add one binary slack variable to (39) because the maximum difference between the right-hand side and the left-hand side of constraint (56) is 1. We transform (57) and (58) to the equations shown as

$$P_{3ijm}\cdot x_{ijm}^2 + P_{5ijmt}\cdot y_{ijmt}^2 + P_{6ijmt}\cdot z_{ijmt}^2 + P_{7ijmt}\cdot p_{ijmt}^2, \quad \forall i,j,t; \quad m \notin V_{ij}^k, \tag{40}$$

$$P_{2ij}\left(\sum_{m\in V_{ij}^k}\sum_{t=1}^{T_{\max}} z_{ijmt} - 1\right)^2 + P_{3ij}\left(\sum_{m\in V_{ij}^k}\sum_{t=1}^{T_{\max}} p_{ijmt} - 1\right)^2, \quad \forall i,j. \tag{41}$$

To form the QUBO formulation of sub-problem 1, all terms in equations (31)-(41) need to be integrated into the right-hand side of (23). The new model of sub-problem 1 denotes as Z_{IP1}^*. All constraints of the second sub-problem are also transformed into penalty terms. Here are the transformation results. We transform (8) to the equation shown as

$$P_{1ijn}\left(\sum_{m=1}^{N} u_{ij(m,n)} - \sum_{m'=1}^{N} u_{i(j+1)(n,m')}\right)^2, \quad \forall i,j; \quad n \in V_{i(j+1)}^{k'}. \tag{42}$$

Eq. (42) has the same effect as (8) because if there is a case that does not satisfy (8), the penalty term in (42) will affect the objective function. Thus, to minimize the objective function, the solver is inclined to steer clear of any case that would trigger this penalty, thereby enforcing compliance with the conditions outlined in (8). We transform (9) and (10) to the equations shown as

$$P_{2ij}\left(\sum_{m\in V_{ij}^k}\sum_{n\in V_{i(j+1)}^{k'}} u_{ij(m,n)} - 1\right)^2, \quad \forall i,j, \tag{43}$$

$$P_{1ijmnt}\Big(v_{ij(m,n)t}^2 - u_{ij(m,n)}v_{ij(m,n)t}\Big), \quad \forall i,j,m,n,t. \tag{44}$$

Eq. (9) is transformed into (43) following the same idea as the transformation of constraint (48). Eq. (44) obviously has the same effect as constraint (10). It specifically avoids the scenarios that there exits y_{ijmt} equals 1 while no x_{ijm} is equal to 1. Constraint (11) is transformed to the equation shown as

$$P_{mnt}\left(\sum_{(i\neq i^*)\vee(j\neq j^*)} v_{ij(m,n)t}\cdot v_{i^*j^*(m,n)t} + \sum_{(i'\neq i^*)\vee(j'\neq j^*)} v_{i'j'(n,m)t}\cdot v_{i^*j^*(n,m)t} + \sum_{(i\neq i')\vee(j\neq j')} v_{ij(m,n)t}\cdot v_{i'j'(n,m)t}\right)^2, \quad \forall m,n,t. \tag{45}$$

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

Algorithm 1 Lagrangian Relaxation based Parallelized Quantum Annealing Algorithm

Require: parameters, I, J, M; the functions in service chain i, f_{ij}^k; the set of VMs which can process f_{ij}^k, V_{ij}^k; the NFV network; the value of penalty coefficients;
Initialize: λ_{ijmn}, λ_{ijmt}, λ_{ijnt}, γ_1, γ_2, γ_3 ;
find a feasible $T_{\max}$;
$Z_{\text{IP}^*} \leftarrow -\infty$;
while $\mid (Z_{\text{IP}}^* - Z_{\text{IP}}^{(p)*})/Z_{\text{IP}}^* \mid \geq \epsilon$ **do**
    solve the QUBO model of subproblem 1 and the QUBO model of subproblem 2 individually by hybrid solvers;
    get Z_{IP1}^*, Z_{IP2}^*, Z_{IP}^*, x_{ijm}, y_{ijmt}, z_{ijmt}, p_{ijmt}, u_{ijmn}, and v_{ijmnt};
    update λ_{ijnt}, λ_{ijmt}, and λ_{ijmn} by (25) to (30);
end while
output Z_{IP}, x_{ijm}, y_{ijmt}, z_{ijmt}, p_{ijmt}, u_{ijmn}, v_{ijmnt};
find the neighborhood of the current solution;
search the possible optimal solution and update the tabu list;
reach the optimal solution;
return Z_{IP}, x_{ijm}, y_{ijmt}, z_{ijmt}, p_{ijmt}, u_{ijmn}, v_{ijmnt}.

We transform constraint (11) to (45) following the same principle used in the transformation of constraint (50). Any $v_{ij(m,n)t}$ or $v_{ij(n,m)t}$ is set as 1, and then others cannot be equal to 1. Otherwise, these penalty terms in (45) will lead the objective function away from the optimal solution. We transform (12) to the equations shown as

$$P_{1ijmn}\left(\sum_{t=1}^{T_{\max}} v_{ij(m,n)t} - T_{ij(m,n)} \cdot u_{ij(m,n)}\right)^2, \quad \forall i,j; \quad m \in V_{ij}^k, \quad n \in V_{i(j+1)}^{k'}. \tag{46}$$

Eq. (46) is equivalent to constraint (12). If the values of $v_{ij(m,n)t}$ and $u_{ij(m,n)}$ cannot satisfy the constraint (12), (46) will add a big value to the objective function so the final solution cannot be minimum. Thus, (46) has the same effects as the constraint (12). Constraint (15) is transformed to the equation shown as

$$P_{2ijmn} \cdot u_{ij(m,n)}^2 + P_{3ijmnt} \cdot v_{ij(m,n)t}^2, \quad \forall i,j,t; \quad m \notin V_{ij}^k, \quad n \notin V_{i(j+1)}^{k'}. \tag{47}$$

If and only if the $v_{ij(m,n)t}$ and $u_{ij(m,n)}$ where $m \in V_{ij}^k$ and $n \in V_{i(j+1)}^{k'}$, equal to 1, (47) will lead the objective function to the optimal solution. Thus, (47) has the same function as the constraint (15). All terms in equations (42)-(47) need to be added to the right-hand side of (24) to formulate the QUBO model of sub-problem 2. These penalty terms maintain the integrity of the original constraints within the transformed model. This new model of sub-problem 2 denotes as Z_{IP2}^* and the whole QUBO model denotes as Z_{IP}^*.

D. ALGORITHM

The framework of the proposed Lagrangian relaxation based parallelized quantum annealing algorithm is shown in Algorithm 1, which is used to solve the VNFs scheduling problem in this paper. To solve this optimization problem, we first build the ILP model and then decompose it by Lagrangian relaxations to form two sub-problems, which are explained at great length in Section IV. We transfer these models to QUBO forms and then use quantum annealing to solve these subproblems separately. Quantum computers will return their optimal solutions to the classical computer. The variables' values in optimal solutions are used to update Lagrange multipliers λ_{ijmn}, λ_{ijmt}, and λ_{ijnt}. If the solutions cannot meet the stop criterion, these updated multipliers will be used to solve sub-problem models again. Finally, the solutions after several iterations satisfy the stop criterion and the loop ends to output an approximal optimal solution to the original model. There are steps of tabu search that can generate the neighborhood of the current solution and then search for the optimal solution to the original problem model.

IV. EXPERIMENT

In this section, we verify the feasibility of the proposed Lagrangian relaxation based parallelized quantum annealing algorithm and analyze its advantages in solving VNFs scheduling problems under many cases.

A. QUANTUM COMPUTING IMPLEMENTATION

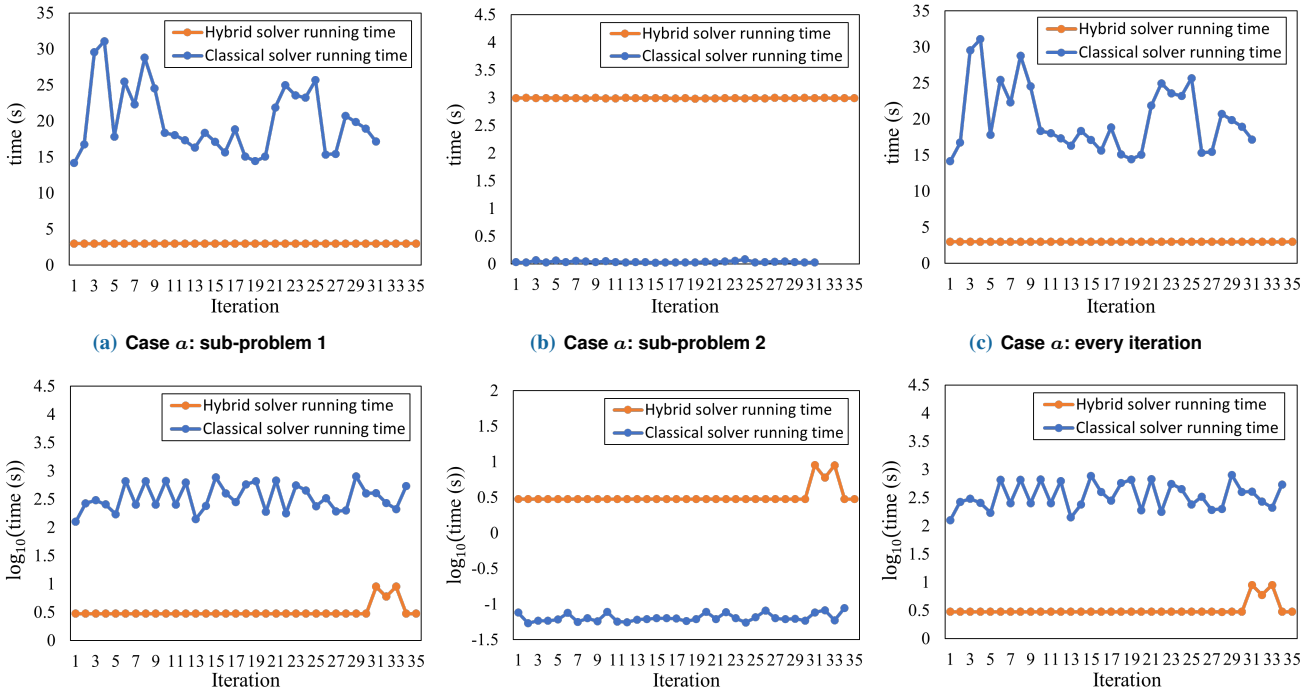
The whole proposed algorithm and Lagrangian relaxation-based classical algorithm run in the Python 3.8 environment. The quantum annealing part of our proposed algorithm was operated on the D-Wave quantum annealers by hybrid solvers named *hybrid_binary_quadratic_model_version2*. These hybrid solvers preprocessed the uploaded QUBO models and leveraged quantum annealers to solve them. The hybrid solver can tackle the model with up to $1,000,000$ variables and employ over $5,000$ qubits. The quantum process units (QPU) have the topology with the Pegasus graph. The QUBO models were embedded on QPU through corresponding packages in D-Wave ocean software to automatically use minor embedding and it can simplify the operations of using quantum annealing. The two sub-problems, formed by the Lagrangian relaxation method shown in Section IV, were solved by two independent hybrid solvers simultaneously to shorten the simulation time further. Compared with the proposed hybrid algorithm, the Lagrangian relaxation-based classical algorithm replaced the hybrid solvers with the Cplex solvers.

B. SIMULATION SETUP

To show the performance of the proposed Lagrangian relaxation based parallelized quantum annealing algorithm, we randomly set different parameters I, J, and M, and different service chains to generate different cases. These parameters are used to provide an expected $T_{\max}$ by a greedy algorithm. Lagrange multipliers λ_{ijnt}, λ_{ijmt} and λ_{ijmn} are set to 1 initialy. In practice, the transmitting rate of data is

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

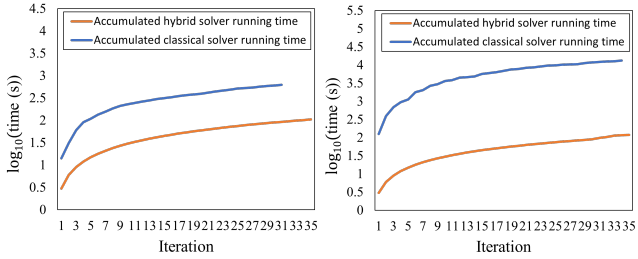
This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433



(a) Case a: sub-problem 1 (b) Case a: sub-problem 2 (c) Case a: every iteration

(d) Case b: sub-problem 1 (e) Case b: sub-problem 2 (f) Case b: every iteration

FIGURE 4. The solver running time of solving sub-problem 1 (a) and sub-problem 2 (b) of case a. (c) is the solver running time per iteration. The solver running time of solving sub-problem 1 (d) and sub-problem 2 (e) of case b. (f) is the solver running time per iteration.



(a) Case a (b) Case b

FIGURE 5. The accumulated solver running time of solving cases a and b.

relatively small so the time delay of transmitting is set as one time slot in all cases. The assumption helps reduce the complexity of QUBO models. We randomly set the value of the workload of all functions in service chains to objectively evaluate the general performance of the proposed algorithm. The penalty coefficients of QUBO models are set to $1,000$ times larger than the solutions. Different penalty terms have different penalty coefficients depending on the situations of corresponding constraints. The hybrid solver may solve the sub-problem models several times to achieve the optimal solution in some iterations. Furthermore, to prevent over-convergence while running these two algorithms, the stop criterion starts working after 30 iterations.

C. SIMULATION RESULT

In this subsection, we investigate the simulation results of the VNFs scheduling problem in many cases. The data presented in Fig. 4 compares the running time of hybrid solvers, which incorporate quantum annealing, against that of Cplex solvers in two distinct cases. For sub-problem 1, it is obviously shown that hybrid solvers obtain the solutions in a much shorter time per iteration compared with classical solvers. These results verify that quantum annealing has superiority while solving complex QUBO models. Based on our observations, the QUBO models in cases a and b are complex for Cplex solvers so Cplex solvers may perform several restarting to reach the optimal solution in iterations. It results in significantly varying classical solver running times per iteration. For sub-problem 2, the classical solvers can get the solution faster than hybrid solvers, which is because the preprocessing procedure consumes a certain amount of time while using hybrid solvers. In case b, the hybrid solver may not reach the optimal solution of sub-problem 2 models, so the solver is employed several times in some iterations. The solver running time per iteration depends on the longer solver running time between solving sub-problem 1 and sub-problem 2 so that the running time of hybrid solvers is shorter than classical solvers per iteration. The classical solver running time per iteration is contributed by the time of solving sub-problem 1. The accumulated solver running time of solving cases a and b is shown in Fig. 5. It is significantly presented that the total classical solver running time is much higher than the hybrid solver running time.

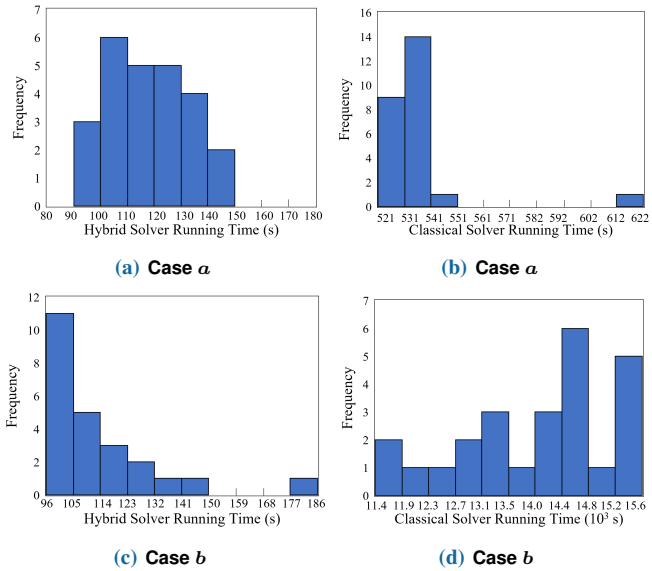
In Table 2, the solver running time of two algorithms in 6 cases are shown in detail. The first column shows the parameter setting of $(I, J, M, T_{\max})$. The second column

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

TABLE 2. Time Consuming

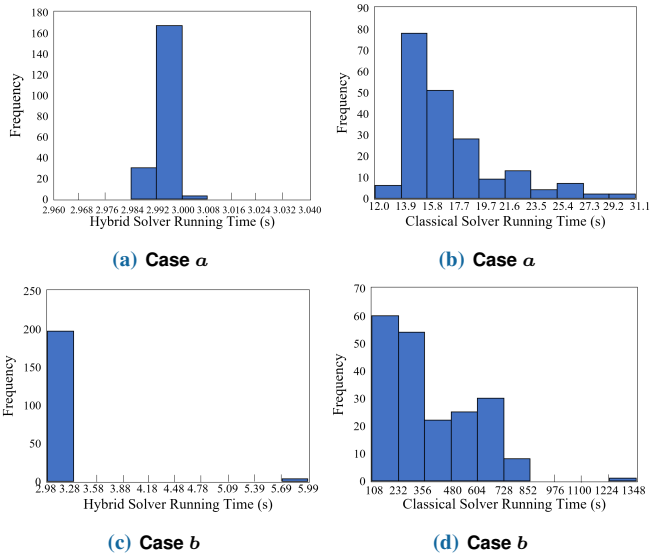
Case	Parameter	Hybrid Solver Run Time (s)	Classical Solver Run Time (s)
c	$(2, 2, 2, 6)$	116.83	664.40
d	$(2, 2, 2, 7)$	104.83	621.92
e	$(2, 2, 3, 8)$	119.81	$13,204.85$
f	$(2, 3, 2, 8)$	248.55	$> 150,000.00$
g	$(3, 2, 2, 10)$	182.68	$> 150,000.00$
h	$(2, 3, 3, 11)$	374.41	$> 150,000.00$



(a) Case a (b) Case a (c) Case b (d) Case b

FIGURE 6. Histograms of the total running time for hybrid solvers and classical solvers in case a and b.

and the third column present the solver running times in the whole iteral procedure by hybrid solvers and classical solvers. The last column shows the gain of hybrid solvers over classical solvers. In all listed cases, hybrid solvers present significant advantages in solving time. For case c, the hybrid solver running time of the whole process is 116.83s but the classical solver running time is up to 664.4s, which shows that our proposed algorithm has significant advantages. For other cases, which are with larger QUBO sizes and more complex formulations, the hybrid solver can achieve much more superiority. Under cases f, g, and h, for every iteration, the classical solver cannot find the optimal solution to subproblem 1 in a reasonable time. It is because the model in these cases is too complex for the classical solver and it gets stuck in the search process. However, these cases are much easier for hybrid solvers to achieve optimal solutions in a short time. These results present our proposed algorithm can solve large-scale VNFs scheduling problems much faster than the Lagrangian relaxation-based classical algorithm.



(a) Case a (b) Case a (c) Case b (d) Case b

FIGURE 7. Histograms of the solver running time per iteration in case a and b.

Fig. 6 showcases the performance of our proposed algorithm versus a classical algorithm based on Lagrangian relaxation through the analysis of simulation results across 25 attempts for each algorithm. In case a, the hybrid solver running times for all iterations fall within the range of 90s to 150s. The distribution of hybrid solver running time for all iterations concentrates in the range of 521s to 551s, but some results are around 617s, which is much higher than others. This noted deviation suggests classical solvers may meet occasional challenges in reaching the optimal solution. In case b, most simulation results using hybrid solvers obtain the solver running time between 96s and 132s. However, the classical solver results are dispersed and distributed between $11,400$s and $15,600$s. It signifies that hybrid solvers are much more stable in solving the VNFs scheduling problems in this scenario. The analysis of 200 iterations results from the hybrid solver and the classical solver are shown in Fig. 7. It is found that the running times of hybrid solvers present more concentrated distributions in these two cases. For some iterations, hybrid solvers are employed more than one time because hybrid solvers fail to achieve the optimal solutions sometimes. From Figs. 6 and 7, it can be seen that our proposed hybrid algorithm shows better performance in robustness.

V. CONCLUSION

In this paper, we proposed a Lagrangian relaxation based parallelized quantum annealing algorithm, designed to tackle complex optimization problems by effectively breaking them down into smaller and more manageable sub-problems. The success of our proposed hybrid algorithm in addressing the VNFs scheduling problem illustrates its applicability and effectiveness in solving large-scale optimization problems. From the results of the case study, we can find that the

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

time performance of the proposed algorithm is better in all cases compared with the Lagrangian relaxation-based classical algorithm. For cases with relatively fewer variables, hybrid solvers spend 116.83s to solve the models but classical solvers spend 5.69 times more than hybrid solvers. Hybrid solvers can reach the solutions to some relatively large-scale cases in 374.41s but these cases are unsolvable for classical solvers in a certain time. Furthermore, our proposed algorithm shows excellent advantages in the aspect of robustness. Based on these experimental results, this paper demonstrates that the Lagrangian relaxation based parallelized quantum annealing algorithm can effectively solve complex ILP models that cannot be solved by the algorithm proposed in [3]. This advantage stems from the fact that the parallelized quantum annealing algorithm avoids the limitations imposed by a restricted number of qubits.

REFERENCES

[1] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6/7, pp. 467–488, Jun. 1982.

[2] D. P. Nazareth and J. D. Spaans, "First application of quantum annealing to IMRT beamlet intensity optimization," *Physics in Medicine & Biology*, vol. 60, no. 10, pp. 4137–4148, May 2015.

[3] W. Xuan, Z. Zhao, L. Fan, and Z. Han, "Minimizing delay in network function visualization with quantum computing," in *IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, Online, Dec. 2021, pp. 108–116.

[4] R. Orus, S. Mugel, and E. Lizaso, "Forecasting financial crashes with quantum computing," *Physical Review A*, vol. 99, no. 6, no. of article 060301, Jun. 2019.

[5] R. Orús, S. Mugel, and E. Lizaso, "Quantum computing for finance: Overview and prospects," *Reviews in Physics*, vol. 4, no. of article 100028, Nov. 2019.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, Sep. 2017.

[7] M. Li, H. Zhang, L. Fan, and Z. Han, "A quantum feature selection method for network intrusion detection," in *IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, Denver, CO, Oct. 2022, pp. 281–289.

[8] Y. Cao, S. Jiang, D. Perouli, and S. Kais, "Solving set cover with pairs problem using quantum annealing," *Scientific Reports*, vol. 6, no. of article 33957, Sep. 2016.

[9] E. Pelofske, G. Hahn, and H. Djidjev, "Decomposition algorithms for solving NP-hard problems on a quantum annealer," *Journal of Signal Processing Systems*, vol. 93, pp. 405–420, Apr. 2021.

[10] K. Ikeda, Y. Nakamura, and T. S. Humble, "Application of quantum annealing to nurse scheduling problem," *Scientific Reports*, vol. 9, no. of article 12837, Sep. 2019.

[11] D. Alanis, P. Botsinis, Z. Babar, H. V. Nguyen, D. Chandra, S. X. Ng, and L. Hanzo, "A quantum-search-aided dynamic programming framework for Pareto optimal routing in wireless multihop networks," *IEEE Transactions on Communications*, vol. 66, no. 8, pp. 3485–3500, Aug. 2018.

[12] F. Feng, P. Zhang, M. A. Bragin, and Y. Zhou, "Novel resolution of unit commitment problems through quantum surrogate lagrangian relaxation," *IEEE Transactions on Power Systems*, vol. 38, no. 3, pp. 2460–2471, May 2023.

[13] Z. Zhao, L. Fan, and Z. Han, "Hybrid quantum benders' decomposition for mixed-integer linear programming," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Austin, TX, Apr. 2022, pp. 2536–2540.

[14] Z. Zhao, L. Fan, and Z. Han, "Optimal data center energy management with hybrid quantum-classical multi-cuts Benders' decomposition method," *IEEE Transactions on Sustainable Energy*, vol. 15, no. 2, pp. 847–858, Apr. 2024.

[15] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: state-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 1st Quart. 2016.

[16] B. Yi, X. Wang, K. Li, and M. Huang, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, Mar. 2018.

[17] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

[18] M. Bagaa, T. Taleb, and A. Ksentini, "Service-aware network function placement for efficient traffic handling in carrier cloud," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Istanbul, Turkey, Apr. 2014, pp. 2402–2407.

[19] M. Gaggero and L. Caviglione, "Model predictive control for energy-efficient, quality-aware, and secure virtual machine placement," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 420–432, Jan. 2019.

[20] T. Albash and D. A. Lidar, "Demonstration of a scaling advantage for a quantum annealer over simulated annealing," *Physical Review X*, vol. 8, no. 3, no.of article 031016, July 2018.

[21] V. Choi, "Minor-embedding in adiabatic quantum computation: I. the parameter setting problem," *Quantum Information Processing*, vol. 7, pp. 193–209, Sep. 2008.

[22] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," *arXiv preprint quant-ph/0001106*, Jan. 2000.

[23] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, no. 1, pp. 1–18, Jan. 1981.

[24] L. Liu, S. Guo, G. Liu, and Y. Yang, "Joint dynamical VNF placement and SFC routing in NFV-enabled SDNs," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4263–4276, Dec. 2021.

[25] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Management science*, vol. 50, no. 12_supplement, pp. 1861–1871, Dec. 2004.

[26] C.-N. Fiechter, "A parallel tabu search algorithm for large traveling salesman problems," *Discrete Applied Mathematics*, vol. 51, no. 3, pp. 243–267, Jul. 1994.

[27] F. Glover, "Tabu search—part II," *ORSA Journal on computing*, vol. 2, no. 1, pp. 4–32, Feb. 1990.

[28] N. Ghaffarinasab, "A tabu search heuristic for the bi-objective star hub location problem," *International Journal of Management Science and Engineering Management*, vol. 15, no. 3, pp. 213–225, 2020.

Appendix A

The settings of VMs will influence the arrangement of scheduling virtual functions to process users' data, so the following constraints are used to represent the characteristics of VMs and to prevent unreasonable arrangements. x_{ijm}, y_{ijmt}, z_{ijmt} and p_{ijmt} are binary decision variables in these constraints. One of these constraints is shown as

$$\sum_{m \in V_{ij}^k} x_{ijm} = 1, \quad \forall i, j. \tag{48}$$

V_{ij}^k denotes the set of VMs that can allow function f_{ij}^k runs on it. If x_{ijm} is equal to 1, it means that the function f_{ij}^k is arranged on VM m to operate. Constraint (48) shows that only one VM that belongs to V_{ij}^k can be used to process function f_{ij}^k. One other constraint is shown as

$$x_{ijm} = \sum_{t=1}^{T_{\max}} z_{ijmt}, \quad \forall i, j; \quad m \in V_{ij}^k \tag{49}$$

z_{ijmt} denotes when the function f_{ij}^k starts to be processed on VM m. Constraint (49) limits that if and only if VM m is

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

employed to process the function f_{ij}^k, there will be a starting time of process function f_{ij}^k on VM m. One other constraint is shown as

$$\sum_{i=1}^{I}\sum_{j=1}^{J} y_{ijmt} \leq 1, \quad \forall m, t. \tag{50}$$

y_{ijmt} denotes that, at the time slot t, VM m is processing the function f_{ij}^k. Constraint (50) represents that VM m can process only one function at the same time. One other constraint is shown as

$$(1 - x_{ijm}) \cdot y_{ijmt} = 0, \quad \forall i, j, m, t. \tag{51}$$

Constraint (51) shows that if and only if VM m is arranged to process the function f_{ij}^k, VM m can process it in time slot t. One other constraint is shown as

$$\sum_{t=1}^{T_{\max}} y_{ijmt} = T_{ijm} \cdot x_{ijm}, \quad \forall i, j; \quad m \in V_{ij}^k. \tag{52}$$

Constraint (52) describes that if VM m is chosen to serve the function f_{ij}^k, it must work for f_{ij}^k in the required time T_{ijm}. One other constraint is shown as

$$z_{ijmt} \cdot p_{ijmt} = 0, \quad \forall i, j, m, t. \tag{53}$$

p_{ijmt} denotes that VM m finishes processing the function f_{ij}^k before the beginning of the time slot t. Constraint (53) represents the mutually exclusive relationship of z_{ijmt} and p_{ijmt} in the same time slot. One other constraint is shown as

$$\begin{aligned} y_{ijm(t-1)} - y_{ijmt} + z_{ijmt} - p_{ijmt} = 0, \\ \forall i, j, t; \quad m \in V_{ij}^k. \end{aligned} \tag{54}$$

Constraint (54) forces time-dependent decision variables, z_{ijmt}, y_{ijmt}, and p_{ijmt}, must follow the logical order. One other constraint is shown as

$$\sum_{\alpha=1}^{T_{ijm}} z_{ijm(t-\alpha+1)} \leq y_{ijmt}, \quad \forall i, j, t; \quad m \in V_{ij}^k. \tag{55}$$

Constraint (55) uses z_{ijmt} and y_{ijmt} to makes sure that VM m process the function f_{ij}^k for enough long time. One other constraint is shown as

$$\begin{aligned} \sum_{m \in V_{ij}^k} \sum_{\beta=1}^{T_{\max}} p_{ijm(t-\beta+1)} \geq z_{i(j+1)m't}, \\ \forall i, j, t; \quad m' \in V_{i(j+1)}^{k'}. \end{aligned} \tag{56}$$

Constraint (56) presents that after finishing processing the function f_{ij}^k, $f_{i(j+1)}^k$ will be processed in some time slots. Other constraints are shown as

$$\begin{aligned} x_{ijm} = y_{ijmt} = z_{ijmt} = p_{ijmt} = 0, \\ \forall i, j, t; \quad m \notin V_{ij}^k, \end{aligned} \tag{57}$$

$$\sum_{m \in V_{ij}^k} \sum_{t=1}^{T_{\max}} z_{ijmt} = \sum_{m \in V_{ij}^k} \sum_{t=1}^{T_{\max}} p_{ijmt} = 1, \quad \forall i, j. \tag{58}$$

Constraint (57) guarantees that all decision variables related to VM m which cannot provide corresponding virtual functions, cannot equal to 1. Constraint (58) shows that VMs can only start processing the function f_{ij}^k for one time and also only finish processing it for one time. The detailed explanations of constraint (48)-(58) are shown in [3].

WENLU XUAN (Student Member, IEEE) received the Bachelor of Engineering degree in microelectronics science and engineering from Southern University of Science and Technology, China, in 2019. She is currently a Ph.D. student in electrical engineering at the University of Houston, TX, USA. Her research interests span a diverse range of topics, including quantum computing, optimization algorithms, game theory, and advanced communication network systems.

ZHONGQI ZHAO (Student Member, IEEE) received a B.S. degree in electronic engineering from Beijing Jiaotong University, in 2018, a B.S. degree in Mathematics from the University of Minnesota, Twin Cities, in 2018, and B.S. and M.S. degrees in electrical engineering from the University of Minnesota, Twin Cities, in 2018 and 2020, respectively. Zhongqi Zhao is currently a Ph.D. student at the University of Houston. His research interests include quantum computing, optimization methods, complex system operations, power system operations, and planning.

LEI FAN (Senior Member, IEEE) is an Assistant Professor in the Department of Engineering Technology and also holds a joint appointment with the Department of Electrical and Computer Engineering at the University of Houston. Prior to this position, he worked in the power industry for several years. He earned his Ph.D. in operations research from the Department of Industrial and Systems Engineering at the University of Florida. His research interests encompass power system operations and planning, electricity markets, optimization algorithms, complex network systems, and quantum computing.

ZHU HAN (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was an R&D Engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a Research Associate at the University of Maryland. From 2006 to 2008, he was an assistant professor at Boise State University, Idaho. Currently, he is a John and Rebecca Moores Professor in the Electrical and Computer Engineering Department as well as in the Computer Science Department at the University of Houston, Texas. Dr. Han's main research targets on the novel game-theory related concepts critical to enabling efficient and distributive use of wireless networks with limited resources.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2024.3424433

His other research interests include wireless resource allocation and management, wireless communications and networking, quantum computing, data science, smart grid, carbon neutralization, security and privacy. Dr. Han received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the Journal on Advances in Signal Processing in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, IEEE Vehicular Technology Society 2022 Best Land Transportation Paper Award, and several best paper awards in IEEE conferences. Dr. Han was an IEEE Communications Society Distinguished Lecturer from 2015 to 2018 and ACM Distinguished Speaker from 2022 to 2025, AAAS fellow since 2019, and ACM Fellow since 2024. Dr. Han is a 1% highly cited researcher since 2017 according to Web of Science. Dr. Han is also the winner of the 2021 IEEE Kiyo Tomiyasu Award (an IEEE Field Award), for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: "for contributions to game theory and distributed management of autonomous communication networks."

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/