Title	An efficient auction mechanism for service chains in the NFV market
Author(s)	Gu, S; Li, Z; Wu, C; Huang, C
Citation	The 35th Annual IEEE International Conference on Computer Communications (IEEE INFOCOM 2016), San Francisco, CA., 10-14 April 2016. In IEEE Infocom Proceedings, 2016, p. 1-9
Issued Date	2016
URL	http://hdl.handle.net/10722/229713
Rights	IEEE Infocom Proceedings. Copyright © IEEE Computer Society.; ©2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.; This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

An Efficient Auction Mechanism for Service Chains in The NFV Market

Sijia Gu[§†], Zongpeng Li[§†], Chuan Wu[‡], Chuanhe Huang[§]
[§] State Key Lab of Software Engineering, School of Computer, Wuhan University
[†] Department of Computer Science, University of Calgary, {*sijia.gu, zongpeng*}*@ucalgary.ca*
[‡] Department of Computer Science, The University of Hong Kong, *cwu@cs.hku.hk*

***Abstract*—Network Function Virtualization (NFV) is emerging as a new paradigm for providing elastic network functions through flexible virtual network function (VNF) instances executed on virtualized computing platforms exemplified by cloud datacenters. In the new NFV market, well defined VNF instances each realize an atomic function that can be chained to meet user demands in practice. This work studies the dynamic market mechanism design for the transaction of VNF service chains in the NFV market, to help relinquish the full power of NFV. Combining the techniques of primal-dual approximation algorithm design with Myerson's characterization of truthful mechanisms, we design a VNF chain auction that runs efficiently in polynomial time, guarantees truthfulness, and achieves near-optimal social welfare in the NFV eco-system. Extensive simulation studies verify the efficacy of our auction mechanism.**

I. INTRODUCTION

Network Function Virtualization (NFV) is emerging as a new network architecture that uses standard IT virtualization techniques to consolidate many network equipment types onto industry standard high volume servers [1]. In a traditional network, each distinct function was typically implemented as a specialized appliance based on proprietary hardware. Such appliances invariably include a substantial amount of software, but the software and hardware cannot be separated. NFV replaces dedicated network devices with software running on general-purpose CPUs or virtual machines. A network function such as firewall, load-balancer, or router deployed in the NFV environment is known as a *virtual network function (VNF)*. The mission of NFV includes rapid service innovation, improved operational efficiencies, reduced power usage, standard and open interfaces, greater flexibility and improved capital efficiencies [2].

Prior to the NFV era, vendors sell a solution with integrated hardware and software. Due to the separation of hardware and software in NFV, interoperability becomes even more important. Now the hardware and software may come from different vendors, and the functions in the network have to work in concert with one another. Towards this goal, recent initiatives from major industry players start to define standards for the NFV market. For instance, Huawei provides the Open Platform for NFV (OPNFV) aimed to ensure compatibility between NFV solutions from different vendors, and allow network service providers to choose from a variety of suppliers. NFV makes it easy to aggregate network functions into data centers or other centralized locations. However, concentration of VNFs sometimes is not the most effective and least expensive approach. For this target, a service provider should be free to locate VNFs in all possible locations, from data centers to network nodes to the customer premises. For example, RAD recently introduced a dedicated customer-edge Distributed NFV (D-NFV) solution [3].

A service provider in the NFV paradigm implements one or more virtualized network functions (VNFs). A VNF by itself does not necessarily provide a usable product or service to NFV customers. To build more complex services, the key notion of *service chaining* [4] is introduced, where multiple VNFs are used in sequence to deliver a service. More specifically, a set of VNFs is specified and data flow traverses these VNFs in a prescriptive order. A service chain may be unidirectional, bidirectional or hybrid. When all VNFs in a service chain process traffic in the forward direction only, the service chain is unidirectional. When all the VNFs process the traffic from both directions, it is bidirectional. Otherwise, it is hybrid. This work focuses on the unidirectional service chain where flows are forwarded through the ordered VNF sequence in one direction.

Depending on the property of a specific VNF, some types of VNF can be configured once and shared by multiple service chains (*e.g.*, anti-virus functions), while other types of VNF can only be used by one service chain (*e.g.*, a firewall). Furthermore, traffic analysis functions such as Deep Packet Inspection (DPI) do not modify the packet; some VNFs convert one packet format into another (*e.g., video transcoding*), which will change the packet size; some security functions such as a firewall drop incoming packets that violate security principles. Consequently, it is necessary to define the data rate for each hop in a service chain. Given the two salient features of VNF service chains, it is challenging to find an efficient solution to the VNF Orchestration problem and to compute suitable prices in the NFV market.

Auction mechanisms represent a flexible and efficient approach to our resource allocation problem in NFV. Different from simple allocation schemes based on fixed pricing, an auction is economically efficient, automatically discovers the

[0]Work supported in part by: the Natural Sciences and Engineering Research Council of Canada (NSERC); SKLSE, Wuhan University (SKLSE-2015-A-06); NSFC (61373040, 61571335, 61572370,); Ph.D. Programs Foundation of Ministry of Education of China (2012014111073), and Hong Kong RGC (HKU 718513, HKU 17204715, HKBU 210412).

market value of service chains, and assigns resources to users who value them most [5]. Instead of using static provisioning, the service provider dynamically assembles VNFs into a service chain and deploys VNF service chains in the data center according to users' demand. Through competitive bidding, it can achieve a higher social welfare than fixed pricing approaches do. However, the social welfare maximization problem under dynamic resource provisioning is NP-hard [6], hence it is challenging to find an effective algorithm to such optimization, even if truthful bids are given for free.

This work focuses on the NFV market for selling service chains in the data center. The service provider acts as the auctioneer and owns network resources in the data center. It sells network services to the users (bidders) through an auction. Each service chain contains a series of VNFs. Due to CPU, memory and bandwidth limits in the data center, It can only configure a finite number of VNF instances that can be packed into service chains. While some VNFs can be shared among several service chains, however, the degree of sharing should be limited. To guarantee the performance of the VNF instance, we should only allow a limited amount of flows traverse a single VNF instance.

To design an economically efficient auction mechanism solving the resource allocation problem while pursuing the maximum social welfare, we formulate the social welfare maximization problem as an integer linear program (ILP) which is an NP-hard combinatorial optimization problem. Thus, it is unlikely to find an exact algorithm to get the solution in polynomial time. We instead resort to the design of an efficient primal-dual approximate algorithm to compute in polynomial time a close-to-optimal resource allocation solution. Our primal-dual algorithm is carefully designed, so that it simultaneously achieves three goals. First, it is computationally efficient, making the NFV auction amenable for practical implementations. Second, it guarantees a small approximation ratio as compared to the optimal solution, ensuring good social welfare in the auction result. Third, our algorithm ensures that a higher bid can only help with the winning probability of an NFV user, satisfying the *monotone allocation rule* [7] that is essential to the design of an accompanying payment mechanism for together eliciting truthful bids.

Truthfulness is a desirable property in auction mechanism design. It is well-known that the Vickrey-Clarke-Groves (VCG) mechanism can guarantee both truthfulness and social welfare maximization, under the assumption that an optimal algorithm is applied to the underlying social welfare maximization problem [8]. However, the VCG auction coupled with an approximate algorithm loses its truthfulness guarantee. Consequently, exploiting the monotone property of our primal-dual algorithm, we resort to design an alternate payment strategy base on Myerson's celebrated characterization of truthful auctions, making sure that bidding truthful valuations for the service chains constitute a dominant strategy for each NFV user. Our primal-dual approximation algorithm works in concert with the tailored payment strategy, together making an efficient truthful NFV auction mechanism.

In the rest of the paper, previous literature is discussed in Sec. II. We formulate our system model in Sec. III. In Sec. IV, we design an auction mechanism containing an primal-dual algorithm to determine the resource provisioning with a small approximation ratio, and a payment strategy to guarantee truthfulness. We evaluate the performance of our auction mechanism in Sec. V. Sec. VI concludes the paper.

II. RELATED WORK

A number of industry standards on NFV were proposed by ETSI, IETF, as well as the industry sector (including Huawei and Cisco), resulting in a series of recently published white papers [1] [9] [10]. The document from IETF [4] defines a reference architecture and a framework to enforce Service Function Chaining (SFC) with minimum requirements on the physical topology of the network. Our work aims at the market design that facilities service chain transactions, and follows the concepts and frameworks outlined in these white papers.

Bari *et al.* [11] present two solutions to the problem of VNF orchestration. They formulate the problem as an ILP and compute the optimal solution using CPLEX when the network is small. When the network is large, a heuristic is used to compute sub-optimal solutions. Gember *et al.* [12] design and implement a network-aware orchestration layer to deploy middleboxes in the cloud, known as Stratos. The orchestration process is separated into three steps: first, deciding how many VNF instances of each type are needed; second, considering how to place these VNFs inside the cloud; finally, routing the traffic through the service chains. In [13], it is shown that properly placing VNFs in an NFV-enabled packet/optical Data Center is effective to minimize optical/electical/optical (O/E/O) conversion cost for VNF service chaining. A heuristic algorithm is provided to solve the problem for VNF placement. To our knowledge, our work is the first to address NFV resource allocation through an efficient auction mechanism, which strives to guarantee not only computational efficiency and approximate social welfare maximization, but also truthful bidding from NFV users.

Auctions have long been used as a mechanism for allocating resources to competing users. Zamban *et al.* [14] design a truthful auction based on an approximation algorithm for cloud resource allocation, but without proving the performance of the algorithm. Wang *et al.* [15] provide a truthful VM auction based on a greedy allocation algorithm and a well-designed payment method. However, the approximation ratio is large and depends on the number of VMs. Zhang *et al.* [6] focus on dynamic cloud resource provisioning, and design a randomized auction for VM transactions in the cloud market. The well-known VCG auction loses its truthful property when applied with an approximation algorithm to compute resource allocation [16]. Hence the VCG mechanism is unsuitable for our NFV service chain auction where optimal allocation is NP-hard. In [17], Gopinathan *et al.* design a payment scheme which can guarantee the truthfulness for approximation algorithm based on Myerson's principle. While most cloud

auction mechanisms sell separate VMs, the auction designed in this work sells correlated VMs in the form of service chains.

III. SYSTEM MODEL

We consider an auction approach to provisioning resources in the NFV market. There are K types of resources, as exemplified by CPU, memory and bandwidth. The total amount of type $k \in [K]$ resource in the data center is c^k. The service provider supplies V types of VNFs, $v \in [V]$ denotes a type v VNF instance that consumes a p_v^k amount of type k resource for each k, and has a processing capacity of M_v. I users participate in the auction as bidders. Each user $i \in [I]$ submits a bid B_i. Each bid B_i requests a service chain, which can be regarded as a directed path containing V_i types of VNFs and E_i links, along with a bidding price b_i. For the service chain in B_i, $(u, v) \in [E_i]$ is a link from VNF instance u to v. The data rate on link (u, v) is $d_i(u, v)$ and $d_i(u, v) = 0$ if (u, v) is not in service chain i. Let a binary x_i indicates whether user i wins ($x_i = 1$) or not ($x_i = 0$). Let v_i denote the true valuation of user i for the requested service chain. If user i wins, the payment for its bid is p_i. The utility u_i for user i is

$$u_i(B_i, \mathcal{B}_{-i}) = \begin{cases} v_i - p_i & \text{if user } i \text{ wins its service chain} \\ 0 & \text{otherwise} \end{cases}$$

$\mathcal{B}_{-i}$ is the set of all the bids except B_i. Our mechanism design aims to guarantee the *Truthfulness* property:

Definition 1 (Truthfulness)**.** An NFV auction is *truthful* if for any user i, declaring the true valuation of the service chain in B_i maximizes its utility, regardless of other users's bids.

We will design a truthful auction that utilizes an approximation algorithm to compute the NFV resource allocation. Along this direction, we will depend on Myerson's characterization of truthful mechanisms [17]. We know that b_i is the bidding price submitted by bidder i. Let b_{-i} denote the other bidding prices except b_i. $P(b_i)$ is i's winning probability when it submits b_i.

Theorem 1. An auction mechanism is truthful (in expectation) if and only if

1) $P_i(b_i)$ is monotonically non-decreasing in b_i, $\forall i \in [I]$;
2) The payment of bidder i bidding b_i is

$$\mathfrak{p}_i = b_i P_i(b_i) - \int_0^{b_i} P_i(b)\, db, \forall i \in [I]$$

Under truthful bidding, the bidding price b_i in user i's bid B_i equals the true valuation v_i. The social welfare maximization problem is to maximize the sum of service provider's utility (revenue), $\sum_{i\in[I]} \mathfrak{p}_i x_i$, and all the bidders' utilities, $\sum_{i\in[I]}(v_i - \mathfrak{p}_i)x_i = \sum_{i\in[I]}(b_i - \mathfrak{p}_i)x_i$, *i.e.*, to maximize $\sum_{i\in[I]} b_i x_i$ since payments cancel themselves.

In the NFV system, we assume that each type of VNF appears in a service chain at most once, and there is no loop in the VNF chain. This is a reasonable simplification that is in line with most practical application scenarios, and helps present and analyze our NFV auction algorithm. Let $h_{i,v}$ denote whether type v VNF is in B_i ($h_{i,v} = 1$) or not ($h_{i,v} = 0$). We assume that the data rate $d_i(u, v)$ for every link that flows into VNF v does not exceed the processing capacity of type v VNF M_v, meanwhile, the usage of any type of resource for a single bid will not exceed the total resource capacity, *i.e.*, $R_k = max_{i\in[I]} \sum_{v\in[V]} p_v^k h_{i,v} < c^k$.

Some VNF instances can be shared among multiple service chains, for better VNF utilization. For example, an anti-virus function can be configured once and used for each chain that relies upon this functionality [18]. Other types of VNF can not be shared. For instance, a firewall needs specific configurations for every chain request. A binary q_v indicates whether a VNF instance v can be shared ($q_v = 1$) or not ($q_v = 0$)).

After receiving the requirements in all user bids, the service provider can classify the total V types of VNF into subsets. Let V_a denote the total number of VNFs that are requested by the users and can be shared among service chains, $[V_a] = [v \in V | q_v = 1, \sum_{i\in I} h_{i,v} \neq 0]$; Let V_b denote the number of VNFs that are requested by the users but can not be shared among service chains, $[V_b] = [v \in V | q_v = 0, \sum_{i\in I} h_{i,v} \neq 0]$.

Because a VNF instance $v \in [V_a]$ permits cross-chain sharing, the service provider can configure it once and provision to every chain that requires this VNF. A binary variable y_v indicates whether a type of VNF $v \in [V_a]$ is active in the data center ($y_v = 1$) or not ($y_v = 0$). When allocating service chains, we have to make sure the allocation respects resource capacity constraints of the data center and processing capacity constraints of VNF instances. The finite supply of each type of resource translates into the following capacity constraint:

$$\sum_{v\in[V_a]} y_v p_v^k + \sum_{v\in[V_b]} \sum_{i\in[I]} p_v^k h_{i,v} x_i \leq c^k, \forall k \in [K] \tag{1}$$

Each service chain i is a directed path. For every node v in chain i, at most one node is connected to v. The notation $d_i(u, v)$ can be simplified to $d_{i,v}$, encoding the data rate flowing into VNF v in service chain i. Given that a single date rate will not exceed the processing capacity M_v, *i.e.*, $D_v = max_{i\in[I]} d_{i,v} < M_v$, the total amount of traffic passing through the shared VNF v should not exceed its processing capacity. Thus, we have the following constraint:

$$\sum_{i\in[I]} x_i d_{i,v} \leq M_v, \forall v \in [V_a] \tag{2}$$

The variable y_v can be derived from x_i as follows:

$$y_v = \begin{cases} 1 & \text{if } \sum_{i\in[I]} x_i h_{i,v} > 0 \\ 0 & \text{if } \sum_{i\in[I]} x_i h_{i,v} = 0 \end{cases} \tag{3}$$

We convert the above equation into a linear constraint:

$$\sum_{i\in[I]} x_i h_{i,v} \leq N y_v, \forall v \in [V_a] \tag{4}$$

The value of N should be selected sufficiently large [19]. Given that $\sum_{i\in[I]} x_i h_{i,v} \leq \sum_{i\in I} h_{i,v} \leq I$, it is safe to select $N = I$. If $\sum_{i\in[I]} x_i h_{i,v} = 0$ is at the optimum, then the maximization of $\sum_{i\in[I]} b_i x_i$ together with constraint (1) will force y_v to zero.

The social welfare maximization problem is:

$$\text{Maximize} \sum_{i\in[I]} b_i x_i \tag{5}$$

Subject To:

$$\sum_{v\in[V_a]} y_v p_v^k + \sum_{v\in[V_b]}\sum_{i\in[I]} p_v^k h_{i,v} x_i \le c^k, \forall k \in [K], \quad (5a)$$

$$\sum_{i\in[I]} x_i d_{i,v} \le M_v, \forall v \in [V_a], \quad (5b)$$

$$\sum_{i\in[I]} x_i h_{i,v} \le I y_v, \forall v \in [V_a], \quad (5c)$$

$$x_i, y_v \in \{0,1\}, \forall i \in [I], v \in [V_a]. \quad (5d)$$

Theorem 2. The social welfare maximization problem defined in ILP (5) is NP-hard.

Proof. The classic combinatorial optimization problem below, 0-1 knapsack, is known to be NP-hard:

$$\text{Maximize} \sum_{i=1}^{n} v_i x_i$$

$$\text{Subject To:} \sum_{i=1}^{n} w_i x_i \le W, x_i \in \{0,1\}$$

ILP (5) with only one of the constraints (5b) is a 0-1 knapsack problem. The knapsack problem is a special case of ILP (5). Hence the social welfare maximization problem in ILP (5) is NP-hard. □

We consider the LP relaxation of ILP (5) by relaxing (5d) to

$$0 \le x_i \le 1, y_v \ge 0, \forall i \in [I], v \in [V_a]. \quad (5d')$$

Then we introduce dual variable vectors λ, β, γ and ϕ to constraints (5a), (5b), (5c) and (5d') respectively. The dual of ILP (5) can be formulated as the following:

$$\text{Minimize} \sum_{k\in[K]} c^k \lambda_k + \sum_{v\in[V_a]} M_v \beta_v + \sum_{i\in[I]} \phi_i \quad (6)$$

Subject To:

$$\phi_i + \sum_{k\in[K]}\sum_{v\in[V_b]} \lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]} \beta_v d_{i,v} + \sum_{v\in[V_a]} \gamma_v h_{i,v} \ge b_i, \forall i \in [I], \quad (6a)$$

$$\sum_{k\in[K]} \lambda_k p_v^k \ge I\gamma_v, \forall v \in [V_a], \quad (6b)$$

$$\lambda_k \ge 0, \beta_v \ge 0, \gamma_v \ge 0, \phi_i \ge 0, \forall k \in [K], \forall v \in [V_a], \forall i \in [I], \quad (6c)$$

IV. A PRIMAL - DUAL APPROXIMATION ALGORITHM

Based on the assumption that the bids we know are truthful, we first design a polynomial time approximation algorithm based on the primal-dual framework [7] for the social welfare maximization problem in ILP (5). The algorithm can output a feasible solution and guarantee a small approximation ratio. Then we design a payment strategy to work in concert with the approximation algorithm, for making sure that bids submitted by the users are truthful in practice.

TABLE I: Summary of Notations

I	# of service users	K	# of resource types
B_i	user i's bid	u_i	utility of user i
b_i	bidding price of bid i	v_i	true valuation of user i

E_i	# of edges in service chain i
V_i	# of VNF types in service chain i
V_a	# of requested VNF types which can be shared
V_b	# of requested VNF types which can not be shared
c^k	the total amount of type k resource
p_v^k	VNF v's requirement of type k resource
M_v	processing capacity of VNF v
R_k	the max user requirement for type k resource
D_v	the max data rate which flows in VNF v
$d_i(u,v)$	user i's demand of data rate flowing from u to v
$d_{i,v}$	user i's demand of data rate flowing into v
$\mathfrak{p}_i$	payment of user i if winning the bid
$h_{i,v}$	$h_{i,v}$ equals 1 when VNF v in bid i, otherwise 0
q_v	q_v equals 1 when VNF v can be shared, otherwise 0
x_i	x_i equals 1 when bid i wins, otherwise 0
y_v	y_v equals 1 if type v VNF is active, otherwise 0

A. The Primal-Dual Approximation Algorithm for Social Welfare Maximization

Our primal-dual algorithm, as shown in Algorithm 1, iteratively selects the current best bid which has the largest value per unit resource, from the set of users who haven't obtained any VNF chain. The currently selected user μ is appended to the set Λ, which contains users who win their corresponding bids. The types of shared VNF requested in the bid μ are known, thus the value of y_v can be determined along with the update of x_μ. Because a shared VNF can be configured once and used by several service chains, in every iteration after the corresponding y_v is updated to 1, the VNF v will be added to the set Δ that contains shared VNFs already configured. Then VNF v will not be configured again. Meanwhile, the algorithm updates the dual variable vectors λ, β and ϕ in every iteration and assigns $\gamma = 0$ from beginning to the end.

Lines 2-4 in Algorithm 1 initialize primal variables x and y and dual variables λ, β, ϕ and γ. In the `while` loop in lines 5-21, variables x, y, λ, β and ϕ are updated iteratively. The `while` loop has three stop conditions. The condition $\Lambda \neq [I]$ ensures that the loop is executed at most I times. If the `while` terminates because $\Lambda \neq [I]$ is false, every user i can receive its service chain. The second condition $\sum_{k\in[K]} c^k \lambda_k < Ke^{\rho-1}$ and the third condition $\sum_{v\in[V_a]} M_v \beta_v < V_a e^{B-1}$ guarantee the feasibility of constraints (5a) and (5b) respectively.

Line 6 selects the current best bid with the largest value of unit resource $\frac{b_i}{\sum_{k\in[K]}\sum_{v\in[V]} \lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]} \beta_v d_{i,v}}$. Line 7 updates the primal variable x and the set Λ. p is the value of the primal objective function. Line 8 sets dual variable ϕ_μ to b_μ, so that $\sum_\mu \phi_\mu = p$ in all iterations. Lines 9-12 update the dual variable λ with exponential cost to reflect the changes of resource utilization. The `for` loop in lines 13-17 repeatedly check the elements in set $[V_a]\backslash\Delta$ and updates the variable y_v as well as the set Δ. Lines 18-20 are similar to lines 10-12, using exponential cost to update the dual variable β.

Algorithm 1 A Primal-Dual Approximation Algorithm for ILP (5)

Input: $(b_i, p_v^k, h_{i,v}, d_{i,v}, R_k, D_v)$
Output: solution (x, y) for ILP (5), (λ, β, γ, ϕ) for LP (6)

1: // Initialization
2: $\rho = min_{k\in[K]} \frac{c^k}{R_k}; B = min_{v\in[V_a]} \frac{M_v}{D_v}$;
3: $\forall i \in [I], \forall v \in [V_a], \forall k \in [K] : x_i = 0; y_v = 0; \phi_i = 0; \gamma_v = 0; \lambda_k = \frac{1}{c^k}; \beta_v = \frac{1}{M_v}$;
4: $p = 0; \Lambda = \emptyset; \Delta = \emptyset; [V'] = [V_b] \bigcup [V_a]$;
5: **while** $\Lambda \neq [I]$ AND $\sum_{k\in[K]} c^k\lambda_k < Ke^{\rho-1}$ AND $\sum_{v\in[V_a]} M_v\beta_v < V_a e^{B-1}$ **do**
6: $\mu = \arg\max_{i\in I\setminus\Lambda}\{\frac{b_i}{\sum_{k\in[K]}\sum_{v\in[V]}\lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]}\beta_v d_{i,v}}\}$;
7: $x_\mu = 1; \Lambda = \Lambda \bigcup \{\mu\}; p = p + b_\mu$;
8: $\phi_\mu = b_\mu$;
9: $[V'] = [V']\setminus\Delta$;
10: **for all** $k \in [K]$ **do**
11: $\lambda_k = \lambda_k(e^{\rho-1})^{(\sum_{v\in[V']} h_{\mu,v}p_v^k)/(c^k - R_k)}$;
12: **end for**
13: **for all** $v \in [V_a]\setminus\Delta$ **do**
14: **if** $h_{\mu,v} = 1$ **then**
15: $y_v = 1; \Delta = \Delta \bigcup \{v\}$;
16: **end if**
17: **end for**
18: **for all** $v \in [V_a]$ **do**
19: $\beta_v = \beta_v(e^{B-1})^{d_{\mu,v}/(M_v - D_v)}$;
20: **end for**
21: **end while**

B. Solution feasibility for the approximation algorithm

Next we make sure that Algorithm 1 outputs a feasible solution to both the primal ILP (5) and the dual LP (6).

Theorem 3. The output of Algorithm 1 includes a feasible solution to ILP (5).

Proof. In Algorithm 1, primal variables x and y are initialized to 0, and will be updated to 1 only; hence constraint (5d) will not be violated. After x_i is set to 1, the corresponding y_v is set to 1 and removed from the original set $[V_a]$. Once y_v is set to 1, it will never become 0 again, satisfying constraint (5c).

We next check the feasibility of constraint (5a). Assume that the lth iteration is the first time that constraint (5a) is violated while the other constraints are satisfied. Thus, at iteration $l-1$,

$$\lambda_k^{l-1} = \frac{1}{c_k}(e^{\rho-1})^{(\sum_{v\in\Delta^{l-1}} p_v^k + \sum_{v\in[V_b]}\sum_{i\in\Lambda^{l-1}} p_v^k h_{i,v})/(c^k - R_k)},$$

and we have $\sum_{v\in\Delta^{l-1}} p_v^k + \sum_{v\in[V_b]}\sum_{i\in\Lambda^{l-1}} p_v^k h_{i,v} \leq c^k$. While in iteration l,

$$\sum_{v\in[V']^{l-1}} h_{\mu^l,v}p_v^k + \sum_{v\in\Delta^{l-1}} p_v^k + \sum_{v\in[V_b]}\sum_{i\in\Lambda^{l-1}} p_v^k h_{i,v} \geq c^k.$$

Because $c^k > R_k \geq \sum_{v\in[V']^{l-1}} h_{\mu^l,v}p_v^k$,

$$\sum_{v\in\Delta^{l-1}} p_v^k + \sum_{v\in[V_b]}\sum_{i\in\Lambda^{l-1}} p_v^k h_{i,v} \geq c^k - \sum_{v\in[V']^{l-1}} h_{\mu^l,v}p_v^k \geq c^k - R_k.$$

Thus, $(\sum_{v\in\Delta^{l-1}} p_v^k + \sum_{v\in[V_b]}\sum_{i\in\Lambda^{l-1}} p_v^k h_{i,v})/(c^k - R_k) \geq 1$. This leads to $\sum_{k\in[K]} c^k\lambda_k^{l-1} \geq Ke^{\rho-1}$, satisfying the second `while` loop stop condition. The `while` loop will terminate before iteration l starts, and constraint (5a) will not be violated.

The proof of feasibility of constraint (5b) is similar to that of (5a). We assume that round t is the first round to violate constraint (5b). Therefore, $\sum_{i\in\Lambda^{t-1}} d_{i,v} \leq M_v$, while, $d_{\mu^t,v} + \sum_{i\in\Lambda^{t-1}} d_{i,v} \geq M_v$. In round t, $\sum_{i\in\Lambda^{t-1}} d_{i,v} \geq M_v - d_{\mu^t,v} \geq M_v - D_v$. It will result in

$$\sum_{v\in[V_a]} M_v\beta_v^{t-1} = \sum_{v\in[V_a]} M_v\frac{1}{M_v}(e^{B-1})^{\sum_{i\in\Lambda^{t-1}} d_{i,v}/(M_v - D_v)} \geq V_a e^{B-1}.$$

By the third stop condition of the `while` loop, this implies that the previous iteration $t-1$ will be the last iteration of the algorithm. To conclude, the primal-dual algorithm provides us with a feasible output to ILP (5). □

Though the primal solution is always feasible for every iteration, we can not guarantee the dual solution is also feasible. We need to apply the method of *dual fitting* to scale the dual by a suitable factor to ensure its feasibility [20].

Theorem 4. Let (λ, β, γ, ϕ) be the vector of current dual variables which is infeasible for the dual LP (6) at the beginning of iteration τ , then the vector $(\varepsilon f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})\lambda, f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})\beta, \gamma, \phi)$ is a feasible fractional solution to the dual LP (6). $f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1}) = b_{\mu^\tau}/(\sum_{k\in[K]}\sum_{v\in[V]}\lambda_k^{\tau-1}p_v^k h_{\mu^\tau,v} + \sum_{v\in[V_a]}\beta_v^{\tau-1}d_{\mu^\tau,v})$, $\varepsilon = \max_{i\in[I]} \frac{\sum_{k\in[K]}\sum_{v\in[V]} p_v^k h_{i,v}}{\sum_{k\in[K]}\sum_{v\in[V_b]} p_v^k h_{i,v}}$.

Proof. We first check the feasibility for constraint (6b). With $\gamma_v = 0$, dual constraint (6b) becomes $\sum_{k\in[K]} \lambda_k p_v^k \geq 0, \forall v \in [V_a]$ which is always feasible because $\lambda_k \geq 0, \forall k \in [K]$.

$\forall i \in \Lambda$, ϕ_i is assigned to b_i. It is clear that constraint (6a) is always satisfied. Then consider the remaining $i \in [I]\setminus\Lambda$, let $f_i(\lambda, \beta) = \frac{b_i}{\sum_{k\in[K]}\sum_{v\in[V]}\lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]}\beta_v d_{i,v}}$. Because $f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})$ is selected from line 6, we have

$$f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1}) \geq f_i(\lambda, \beta), \forall i \in [I]\setminus\Lambda.$$

By the definition of ε, we can get $\varepsilon\sum_{k\in[K]}\sum_{v\in[V_b]}\lambda_k p_v^k h_{i,v} \geq \sum_{k\in[K]}\sum_{v\in[V]}\lambda_k p_v^k h_{i,v}$. Because $\gamma_v = 0, \forall v \in [V_a]$ and we have $\phi_i = 0, \forall i \in [I]\setminus\Lambda$, we have,

$$\begin{aligned}
&\phi_i + \varepsilon f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})\sum_{k\in[K]}\sum_{v\in[V_b]}\lambda_k p_v^k h_{i,v} \\
&+ f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})\sum_{v\in[V_a]}\beta_v d_{i,v} + \sum_{v\in[V_a]}\gamma_v h_{i,v} \\
&\geq f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})(\sum_{k\in[K]}\sum_{v\in[V]}\lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]}\beta_v d_{i,v}) \\
&\geq f_i(\lambda, \beta)(\sum_{k\in[K]}\sum_{v\in[V]}\lambda_k p_v^k h_{i,v} + \sum_{v\in[V_a]}\beta_v d_{i,v}) = b_i
\end{aligned} \tag{7}$$

Consequently, the fitted solution $(\varepsilon f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})\lambda$, $f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})\beta$, γ, $\phi)$ is feasible to the dual LP (6). □

After proving the correctness of Algorithm 1, we next analyze its computation complexity, and show that Algorithm 1 can compute a feasible solution in polynomial time.

Theorem 5. The computational complexity of Algorithm 1 is polynomial time.

Proof. The `while` loop in Algorithm 1 has at most I iterations. In the loop body, line 6 is executed at most $2I$ times. The `for` loop in lines 10-12 executes K times in each `while` loop iteration. Meanwhile, the `for` loops in lines 13-17 and lines 18-20 can both be finished in at most V_a iteration steps. Thus, the complexity of Algorithm 1 is $O(I^2)$. □

Theorem 6. Algorithm 1 computes an α-approximate solution to ILP (5), where $\alpha = \max\{\frac{\varphi\varepsilon}{\ln(\frac{Ke^{\rho-1}+V_a}{K+V_a})}, \frac{\varphi\varepsilon}{\ln(\frac{K+V_a e^{B-1}}{K+V_a})}\} + 1$.

Proof. Let ϖ denote the terminating iteration of Algorithm 1. Let $w_1(\tau) = \sum_{i\in[I]}\phi_i^\tau$, $w_2(\tau) = \sum_{k\in[K]} c^k\lambda_k^\tau$, $w_3(\tau) = \sum_{v\in[V_a]} M_v\beta_v^\tau$ at iteration τ. w^* denotes the optimal dual solution to LP (6). We have

$$\begin{aligned}
w_2(\tau) &= \sum_{k\in[K]} c^k\lambda_k^\tau \\
&= \sum_{k\in[K]} c^k\lambda_k^{\tau-1}(e^{\rho-1})^{(\sum_{v\in[V']} h_{\mu^\tau,v}p_v^k)/(c^k-R_k)} \\
&= \sum_{k\in[K]} c^k\lambda_k^{\tau-1}(1+\frac{\delta}{\frac{c^k}{R_k}-1})^{(\sum_{v\in[V']} h_{\mu^\tau,v}p_v^k)/R_k} \\
&\le \sum_{k\in[K]} c^k\lambda_k^{\tau-1}(1+\frac{\delta}{\frac{c^k}{R_k}-1}((\sum_{v\in[V']} h_{\mu^\tau,v}p_v^k)/R_k)) \\
&= \sum_{k\in[K]} c^k\lambda_k^{\tau-1} + \sum_{k\in[K]}\frac{\delta c^k}{c^k-R_k}(\sum_{v\in[V']} h_{\mu^\tau,v}p_v^k)\lambda_k^{\tau-1} \\
&\le w_2(\tau-1) + \sigma\sum_{k\in[K]}\sum_{v\in[V']} h_{\mu^\tau,v}p_v^k\lambda_k^{\tau-1} \\
&\le w_2(\tau-1) + \sigma\sum_{k\in[K]}\sum_{v\in[V]} h_{\mu^\tau,v}p_v^k\lambda_k^{\tau-1}
\end{aligned} \tag{8}$$

Because $(1+a)^x \le 1+ax$ when $0\le x\le 1$, we obtain the first inequality. While $\delta = (\frac{c^k}{R_k}-1)((e^{\rho-1})^{\frac{1}{\frac{c^k}{R_k}-1}}-1)$, $\sigma = max_{k\in[K]}\frac{\delta c^k}{c^k-R_k}$ and $\rho = min_{k\in[K]}\frac{c^k}{R_k}$, we can get

$$\begin{aligned}
\sigma &= max_{k\in[K]}\frac{\delta c^k}{c^k-R_k} = max_{k\in[K]}\frac{c^k}{R_k}((e^{\rho-1})^{\frac{1}{\frac{c^k}{R_k}-1}}-1) \\
&= \rho(e-1)
\end{aligned}$$

The derivation steps for $w_3(\tau)$ are similar to those of $w_2(\tau)$. We define $A = (\frac{M_v}{D_v}-1)((e^{B-1})^{\frac{1}{\frac{M_v}{D_v}-1}}-1)$ and $\theta = max_{v\in[V_a]}\frac{AM_v}{M_v-D_v} = max_{v\in[V_a]}\frac{M_v}{D_v}((e^{B-1})^{\frac{1}{\frac{M_v}{D_v}-1}}-1)$. Because $B = min_{v\in[V_a]}\frac{M_v}{D_v}$ and $\frac{AM_v}{M_v-D_v}$ is non-increasing with $\frac{M_v}{D_v}>1$, we have $\theta = B(e-1)$.

$$\begin{aligned}
w_3(\tau) &= \sum_{v\in[V_a]} M_v\beta_v^\tau \\
&= \sum_{v\in[V_a]} M_v\beta_v^{\tau-1}(e^{B-1})^{d_{\mu^\tau,v}/(M_v-D_v)} \\
&= \sum_{v\in[V_a]} M_v\beta_v^{\tau-1}(1+\frac{A}{\frac{M_v}{D_v}-1})^{d_{\mu^\tau,v}/D_v} \\
&\le \sum_{v\in[V_a]} M_v\beta_v^{\tau-1}(1+\frac{A}{\frac{M_v}{D_v}-1}(d_{\mu^\tau,v}/D_v)) \\
&= \sum_{v\in[V_a]} M_v\beta_v^{\tau-1} + \sum_{v\in[V_a]}\frac{AM_v}{M_v-D_v}d_{\mu^\tau,v}\beta_v^{\tau-1} \\
&\le w_3(\tau-1) + \theta\sum_{v\in[V_a]} d_{\mu^\tau,v}\beta_v^{\tau-1}
\end{aligned} \tag{9}$$

With the definition of $f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})$, we can obtain,

$$\sum_{k\in[K]}\sum_{v\in[V]} h_{\mu^\tau,v}p_v^k\lambda_k^{\tau-1} + \sum_{v\in[V_a]}\beta_v^{\tau-1}d_{\mu^\tau,v} = \frac{b_{\mu^\tau}}{f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})} \tag{10}$$

In Algorithm 1, p^τ represents the solution of primal ILP (5) in round τ, then we can have $p^\tau - p^{\tau-1} = b_{\mu^\tau}$. We define $\varphi = max\{\sigma,\theta\}$. According to Eqn. (8), Eqn. (9) as well as Eqn. (10), we have,

$$\begin{aligned}
w_2(\tau)+w_3(\tau) &\le w_2(\tau-1)+w_3(\tau-1) \\
&\quad + \sigma\sum_{k\in[K]}\sum_{v\in[V]} h_{\mu^\tau,v}p_v^k\lambda_k^{\tau-1} + \theta\sum_{v\in[V_a]} d_{\mu^\tau,v}\beta_v^{\tau-1} \\
&\le w_2(\tau-1)+w_3(\tau-1) \\
&\quad + \varphi(\sum_{k\in[K]}\sum_{v\in[V]} h_{\mu^\tau,v}p_v^k\lambda_k^{\tau-1} + \sum_{v\in[V_a]} d_{\mu^\tau,v}\beta_v^{\tau-1}) \\
&= w_2(\tau-1)+w_3(\tau-1) + \varphi\frac{b_{\mu^\tau}}{f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})} \\
&= w_2(\tau-1)+w_3(\tau-1) + \varphi\frac{p^\tau-p^{\tau-1}}{f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})}
\end{aligned} \tag{11}$$

To analyze the approximation ratio of Algorithm 1, we should separately consider it in different cases as the algorithm may terminate upon different stop conditions. When $\Lambda == [I]$, all users win their bids. The output is optimal, so the approximation ratio is 1.

Next assume the algorithm stops due to $\sum_{k\in[K]} c^k\lambda_k \ge Ke^{\rho-1}$ or $\sum_{v\in[V_a]} M_v\beta_v \ge V_a e^{B-1}$. If at an iteration t before the terminate iteration ϖ, $w_1(t-1)\ge\frac{w^*}{\alpha}$, it means we already have an α-approximate solution at round t as $w_1(t)$ is non-decreasing. Now we need to consider the case that until the terminating round ϖ, $w_1(t-1)<\frac{w^*}{\alpha}$ is always satisfied to continue upper-bounding the value of $w_2(\tau)+w_3(\tau)$.

Theorem 4 indicates that $(\varepsilon f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})\lambda$, $f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})\beta$, γ, $\phi)$ is a feasible solution to the dual LP (6) at round τ. By weak duality, every feasible dual solution is an upper bound of the optimal solution w^*, therefore,

$$\begin{aligned}
w^* &\le w_1(\tau-1) + \varepsilon f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})w_2(\tau-1) \\
&\quad + f_{\mu^\tau}(\lambda^{\tau-1},\beta^{\tau-1})w_3(\tau-1)
\end{aligned}$$

By the definition of ε, we know it is a constant number no less than 1. Furthermore, $\forall \tau \leq \varpi$, $w_1(\tau-1) < \frac{w^*}{\alpha}$. Thus,

$$\begin{aligned} &\frac{1}{f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})} \leq \frac{\varepsilon w_2(\tau-1) + w_3(\tau-1)}{w^* - w_1(\tau-1)} \\ \leq &\frac{\varepsilon(w_2(\tau-1)+w_3(\tau-1))}{w^* - w_1(\tau-1)} \leq \varepsilon \frac{\alpha}{\alpha-1} \frac{w_2(\tau-1)+w_3(\tau-1)}{w^*} \end{aligned} \tag{12}$$

Recall the Eqn. (11) together with the upper bound for $\frac{1}{f_{\mu^\tau}(\lambda^{\tau-1}, \beta^{\tau-1})}$ in Eqn. (12), we can bound $w_2(\varpi) + w_3(\varpi)$:

$$\begin{aligned} &w_2(\varpi) + w_3(\varpi) \leq w_2(\varpi-1) + w_3(\varpi-1) \\ &+ \frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}(p^\varpi - p^{\varpi-1})(w_2(\varpi-1)+w_3(\varpi-1)) \\ &= (w_2(\varpi-1)+w_3(\varpi-1))(1 + \frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}(p^\varpi - p^{\varpi-1})) \\ &\leq (w_2(\varpi-1)+w_3(\varpi-1))\exp(\frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}(p^\varpi - p^{\varpi-1})) \\ &\leq (w_2(0)+w_3(0))\exp(\frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}p^\varpi) \end{aligned} \tag{13}$$

The second inequality in Eqn. (13) is due to $(1+x) \leq e^x, \forall x \geq 0$. The initial values $w_2(0)$ and $w_3(0)$ are K and V_a respectively. When the algorithm terminates upon stop condition $\sum_{k\in[K]} c^k\lambda_k \geq Ke^{\rho-1}$, it indicates that $w_2(\varpi) \geq Ke^{\rho-1}$. And we know $w_3(\tau)$ is non-decreasing, thus, $w_3(\varpi) \geq w_3(0) = V_a$. We have,

$$Ke^{\rho-1} + V_a \leq (K+V_a)\exp(\frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}p^\varpi) \tag{14}$$

By weak LP duality, we know that $\frac{OPT^*}{p^\varpi} \leq \frac{w^*}{p^\varpi}$, where OPT^* represents the optimal objective value for ILP (5). From Eqn. (14), we can obtain the inequality,

$$\frac{w^*}{p^\varpi} \leq \frac{\alpha}{\alpha-1}\frac{\varphi\varepsilon}{\ln(\frac{Ke^{\rho-1}+V_a}{K+V_a})} = \alpha$$

Then we can get the approximation ratio when the algorithm terminates at the stopping condition $\sum_{k\in[K]} c^k\lambda_k \geq Ke^{\rho-1}$

$$\alpha = \frac{\varphi\varepsilon}{\ln(\frac{Ke^{\rho-1}+V_a}{K+V_a})} + 1 \tag{15}$$

Similarly, when the algorithm terminates at the stopping condition $\sum_{v\in[V_a]} M_v\beta_v \geq V_a e^{B-1}$, with $w_3(\varpi) \geq V_a e^{B-1}$ and $w_2(\varpi) \geq K$, we can get

$$K + V_a e^{B-1} \leq (K+V_a)\exp(\frac{\varphi\varepsilon\alpha}{(\alpha-1)w^*}p^\varpi)$$

Under this condition, the approximation ratio is

$$\alpha = \frac{\varphi\varepsilon}{\ln(\frac{K+V_a e^{B-1}}{K+V_a})} + 1 \tag{16}$$

Finally, by summarizing all three cases discussed above, the approximation ratio of Algorithm 1 is $\alpha = \max\{\frac{\varphi\varepsilon}{\ln(\frac{Ke^{\rho-1}+V_a}{K+V_a})}, \frac{\varphi\varepsilon}{\ln(\frac{K+V_a e^{B-1}}{K+V_a})}\} + 1$ □

C. A truthful payment strategy for the social welfare maximization problem

Algorithm 1 provides us a feasible resource allocation solution in polynomial time. Next we need to ensure that our mechanism is able to offer rational payments for every winning bidder, which can guarantee truthfulness. By the second condition of Theorem 1, we need to find a threshold bidding price b_i^* as our mechanism is deterministic and $P_i(b_i)$ is binary: $P_i(b_i)$ is 0 if $b_i < b_i^*$, and is 1 otherwise. Given the threshold bidding price b_i^*, the payment rule is:

$$\mathfrak{p}_i = b_i P_i(b_i) - \int_{b_i^*}^{b_i} P_i(b)\, db = b_i^*, \forall i \in [I]$$

We can compute the threshold bidding price b_i^* using the following Algorithm 2. For each winning bidder, its payment $\mathfrak{p}_i$ returned from Algorithm 2 is upper-bounded by its bidding price b_i.

Algorithm 2 Payment strategy

```
// Initialization
∀i ∈ [I], p_i = 0;
for all i ∈ [I] do
    if x_i == 1 then
        m = 0; n = b_i;
        while (n − m) > ε do
            Run Algorithm 1 with ((m+n)/2, b_{−i}) as bids;
            if bidder i wins then
                n = (m+n)/2;
            else
                m = (m+n)/2;
            end if
        end while
        p_i = (m+n)/2;
    else
        p_i = 0;
    end if
end for
```

Theorem 7. Algorithm 1 (allocation method) together with Algorithm 2 (payment strategy) constitute a truthful auction.

Proof. Leveraging Theorem 1, we first inspect whether our mechanism can satisfy the first condition, *i.e.*, $P_i(b_i)$ is monotonically non-decreasing in b_i, $\forall i \in [I]$. Assume bidder i wins at a time with bidding price b_i while other bidders' bidding price b_{-i} and the amount of resources remain intact. Now bidder i still wins when it increases its bidding price b_i, since the allocation rule in Algorithm 1 is greedy. If the bidding price is lower than b_i, bidder i may lose because other bids may be chosen before b_i. Therefore the winning probability $P_i(b_i)$ is non-decreasing.

As discussed above, we have verified that the winning probability $P_i(b_i)$ is non-decreasing with the bidding price b_i. Next we need to demonstrate our payment strategy obeys the second condition in Theorem 1. The payment $\mathfrak{p}_i$ which is calculated by Algorithm 2 is satisfied $|\mathfrak{p}_i - b_i^*| \leq \epsilon$ where ϵ is a small positive real number given as margin of error. Thus, the threshold bidding price b_i^* can be found by Algorithm 2 which satisfies the second condition in Theorem 1. □

D. Auction Framework for NFV Social Welfare Maximization

Algorithm 3 shows the overall auction framework for our NFV market. The first part of Algorithm 3 is to apply the

primal-dual approximate algorithm to compute the resource allocation solution to determine the winning bids. After that, payments of all the winning bidders are determined by the payment strategy in Algorithm 2.

Algorithm 3 Auction Framework

Input: $(\vec{b}, p_v^k, h_{i,v}, d_{i,v}, R_k, D_v)$
Output: allocation solution $(\vec{x}, \vec{y})$ for ILP (5) and payment $\vec{\mathfrak{p}}$

1: Compute allocation solution ($\vec{x}$, $\vec{y}$) = Algorithm 1 ($\vec{b}$, p_v^k, $h_{i,v}$, $d_{i,v}$, R_k, D_v);
2: Compute payment $\vec{\mathfrak{p}}$ = Algorithm 2 ($\vec{x}$, $\vec{b}$)
3: **return** $\vec{x}$ and $\vec{\mathfrak{p}}$

V. PERFORMANCE EVALUATION

In the simulation studies, we study an NFV service chain market, in which five types of VNF instances can be shared among different users, and other types of VNFs can not be shared — one instance serves one user exclusively. Each VNF instance requires three types of resources: CPU, memory and bandwidth. We generate the bids and the corresponding bidding prices following random distributions.

A. Performance of the approximation algorithm

First, our theoretical analysis proved an approximation ratio for Algorithm 1 with α. Fig. 1 plots the theoretical approximation ratio α with $\varepsilon = 1.5$, $\varphi = \rho(e-1)$, and the algorithm stops when $\sum_{k\in[K]} c^k \lambda_k \geq K e^{\rho-1}$. We can find that with the increasing of ρ, the ratio is always between 3 and 4. Then we execute the Algorithm 1 separately with the number of bidders ranging from 10 to 100. As shown in Fig. 2, the theoretical approximation ratio proven in Theorem. 6 is much larger than the real approximation ratio calculated with the algorithm iteration. The algorithm achieves a satisfying performance with a small approximation ratio between 1 and 2 in practice. Furthermore, we can find that the approximation ratio is relatively stable when the number of bidders increases. The increase in bidder population does not make the approximation ratio worse. Fig. 3 compares the optimal and approximate social welfare. The gap between the two is small with the ratio close to 1, which indicates the algorithm can compute an allocation solution close to the optimal one in practice.

B. Static allocation vs Dynamic allocation

We next compare the resource allocation performance between static allocation and dynamic allocation. For static resource allocation, we assume the service provider offers 10 types of service chains packed with some fixed types of VNFs. Every user specifies one from the 10 fixed types of service chains along with a bidding price in its bid. In Fig. 4, *user satisfaction* is the ratio of the number of winning users to the total number of users. Decreasing trends can be observed in user satisfaction for both dynamic and static resource allocation, with more bidders participating. For a given number of bidders, the user satisfaction for dynamic resource provisioning is higher than that of the static method. Furthermore, for the Social welfare shown in Fig. 5, dynamic allocation also outperforms static allocation.

C. Payment strategy

In Fig. 6, the solid red (top) line represents the optimal solution to the social welfare maximization problem, the solid black (middle) line is the social welfare calculated by the primal-dual approximation algorithm, while the dashed blue (bottom) line is the payment computed by Algorithm 2, the payment strategy. We can see that the trend in social welfare is increasing with a growing bidder population. This is due to the fact that there are more choices for bid selection, which helps achieve higher social welfare. Algorithm 2 is a payment strategy to compute the threshold bidding price b_i^* as the payment for each winning bidder. The threshold bidding price b_i^* should always be smaller than the bidding price b_i submitted by the bidder. This is verified by the fact that the payment (bottom) line is always below the black (middle) line that represents the corresponding social welfare.

D. Level of overall resource supply

We examine the performance of algorithm 1 when the amount of resource provided by the service provider varies. The histogram in Fig. 7 is the social welfares when the users can choose from different types of VNFs. We separately show the social welfare with the conditions $V_b = 5$, $V_b = 10$ as well as $V_b = 15$. It can be seen that the social welfare decreases when the number of VNF increases. During the simulation, the bidding prices as well as the total amount of resources remain the same, however the requirement for each user increases, thus the ratio of winning users decreases. Consequently the social welfare decreases. In Fig. 8, the approximation ratio decreases (becomes better) with the increase of the total amount of resource. It results from more relaxed restriction of constraint (5a), which makes it easier for our approximation algorithm to approach the optimal solution.

VI. CONCLUSION

NFV is emerging as a new paradigm for providing flexible, elastic and cost effective network functions based on software algorithms executed upon common computing platforms. As a key concept in NFV, a service chain is the unit of transaction in the NFV market that connects atomic network functions to provide composite services. This work is the first to design an efficient auction mechanism for the dynamic provisioning and pricing of NFV service chains in a datacenter. The NFV auction we design is computationally efficient, truthful, and achieves near-optimal social welfare. As an interesting future direction, one may study the design of a double auction for NFV markets with multiple VNF suppliers.

REFERENCES

[1] "Network Functions Virtualization - Introductory White Paper," https://portal.etsi.org/nfv/nfv_white_paper.pdf.
[2] "Network Functions Virtualization (NFV); Use Cases," http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.

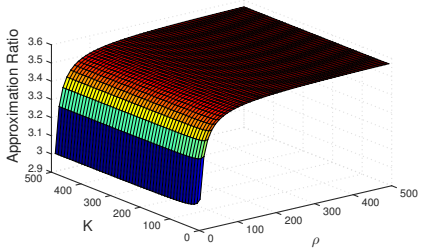

Fig. 1: Theoretical approximation ratio; $\varepsilon = 1.5$, $\varphi = \rho(e-1)$

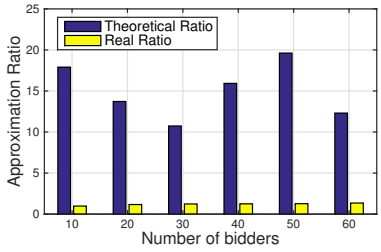

Fig. 2: Comparison between theoretical ratio and real ratio

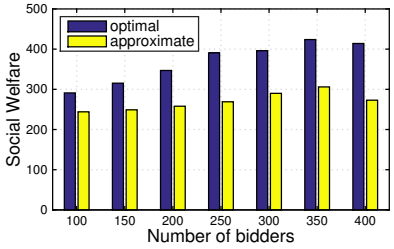

Fig. 3: Comparison between optimal and approximate social welfare

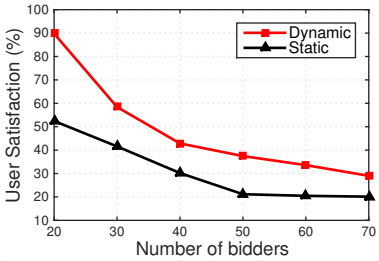

Fig. 4: User Satisfaction for Static allocation and Dynamic allocation

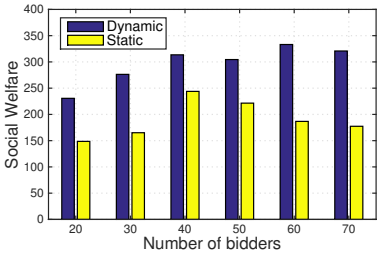

Fig. 5: Social welfare for Static allocation and Dynamic allocation

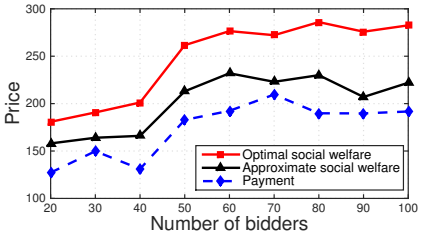

Fig. 6: Performance of payment strategy

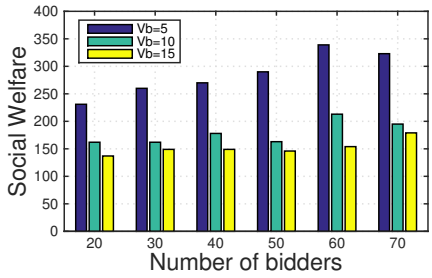

Fig. 7: Social welfare under deferent types of VNFs

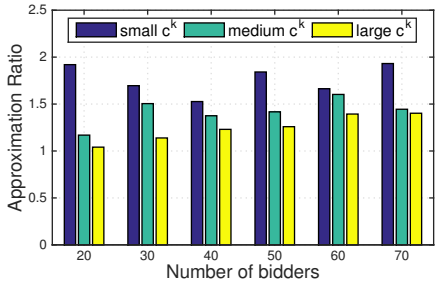

Fig. 8: Comparison of approximation ratio under different amount of resource c^k

[3] "4 Vendors Bring Distributed NFV to BTE," http://www.lightreading.com/4-vendors-bring-distributed-nfv-to-bte/d/d-id/709403.

[4] "Service Function Chaining: Framework & Architecture," https://tools.ietf.org/html/draft-boucadair-sfc-framework-02.

[5] H. Fu, Z. Li, C. Wu, and X. Chu, "Core-selecting auction design for dynamically allocating heterogeneous vms in cloud computing," in *Proc. of IEEE Cloud*, 2014.

[6] L. Zhang, Z. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," in *Proc. of IEEE INFOCOM*, 2014.

[7] P. Briest, P. Krysta, and B. Vöcking, "Approximation techniques for utilitarian mechanism design," in *Proc. of ACM STOC*, 2005.

[8] A. Mu'Alem and N. Nisan, "Truthful approximation mechanisms for restricted combinatorial auctions," *Games and Economic Behavior*, vol. 64, no. 2, pp. 612–631, 2008.

[9] "NFV Management and Orchestration: Enabling Rapid Service Innovation in the Era of Virtualization," http://www.cisco.com/c/en/us/solutions/collateral/service-provider/network-functions-virtualization-nfv/white-paper-c11-732123.pdf.

[10] "White Paper - Huawei Observation to NFV," http://www.huawei.com/ilink/en/download/HW_399662.

[11] M. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba *et al.*, "Orchestrating Virtualized Network Functions," *arXiv:1503.06377*, 2015.

[12] A. Gember, A. Krishnamurthy, S. S. John, R. Grandl, X. Gao, A. Anand, T. Benson, A. Akella, and V. Sekar, "Stratos: A network-aware orchestration layer for middleboxes in the cloud," *arXiv:1305.0209v1*, 2013.

[13] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network Function Placement for NFV Chaining in Packet/Optical Datacenters," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1565–1570, 2014.

[14] S. Zaman and D. Grosu, "Combinatorial auction-based dynamic vm provisioning and allocation in clouds," in *Proc. of IEEE CloudCom*, 2011.

[15] Q. Wang, K. Ren, and X. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in *Proc. of IEEE INFOCOM*, 2012.

[16] N. Nisan and A. Ronen, "Computationally feasible VCG mechanisms," in *Proc. of ACM EC*, 2000.

[17] A. Gopinathan and Z. Li, "Strategyproof auctions for balancing social welfare and fairness in secondary spectrum markets," in *Proc. of IEEE INFOCOM*, 2011.

[18] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. of IEEE CloudNet*, 2014.

[19] H. A. Taha, *Operations Research: An Introduction*. Prentice-Hall, Inc., 2006.

[20] V. V. Vazirani, *Approximation algorithms*. Springer Science & Business Media, 2013.

[21] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[22] R. Lavi and C. Swamy, "Truthful and near-optimal mechanism design via linear programming," *JACM*, vol. 58, no. 6, p. 25, 2011.