PAPER

Incentive-Stable Matching Protocol for Service Chain Placement in Multi-Operator Edge System

Jen-Yu WANG†, **Li-Hsing YEN**†, ***and*** **Juliana LIMAN**†, ***Nonmembers***

SUMMARY Network Function Virtualization (NFV) enables the embedding of Virtualized Network Function (VNF) into commodity servers. A sequence of VNFs can be chained in a particular order to form a service chain (SC). This paper considers placing multiple SCs in a geo-distributed edge system owned by multiple service providers (SPs). For a pair of SC and SP, minimizing the placement cost while meeting a latency constraint is formulated as an integer programming problem. As SC clients and SPs are self-interested, we study the matching between SCs and SPs that respects individual's interests yet maximizes social welfare. The proposed matching approach excludes any blocking individual and block pair which may jeopardize the stability of the result. Simulation results show that the proposed approach performs well in terms of social welfare but is suboptimal concerning the number of placed SCs.
key words: *service chain placement, edge computing, matching*

1. Introduction

Network Function Virtualization (NFV) exploits virtualization technique to embed network function into commodity servers, switches, and storages. It can help reducing the capital expenses (CAPEX), operating expenses (OPEX), and facilitating time-to-market [1] [2]. Network functions implemented as Virtualized Network Functions (VNFs) could be instantiated in virtual machines (VMs) hosted by different physical machines at various locations. Service Function Chaining (SFC) is to chain a sequence of VNFs in a particular order to form a *service chain* (SC). Service chain placement (SCP) is to deploy SCs into a physical or virtualized infrastructure. SCP consists of two tasks. 1) *VNF placement* (also known as Virtual Network Embedding [3]), which is to place VNFs with specific demands in the infrastructure. The goal is usually to minimize the placement cost. 2) *SFC routing*, which is to statically or dynamically determine the route between two consecutive VNFs in an SC. The goal is typically to minimize the latency.

Many SCP approaches assumed cloud data centers as the underlying infrastructure (e.g., [4]). This paper instead considers edge system, which places virtualized computation and storage resource in a location close to end users. Edge system enhances user experience by providing low-latency service. Existing approaches to SCP in edge system aim to minimize overall latency [5], minimize total expected end-to-end latency [6], or jointly minimize the traffic cost and operational cost [7]. Most studies assumed only one network operator or edge service provider (SP). With this assumption, a client, who intends to deploy SCs on an edge system, cannot benefit from choosing a best SP that meets the client's demand yet costs the least. The research in [8] assumed multiple SPs and minimized the overall monetary cost for clients to reach the client-beneficial result. By contrast, our work favors neither clients nor SPs.

We assume that multiple clients want to minimize their payments to SPs for SC placement whereas multiple SPs want to maximize their profits by minimizing their placement costs. In some sense, SPs and clients have conflicting interest since SP's utilities can be increased if clients increase their payments. But doing so will decrease client's utilities. Our goal in this study is to provide a trading platform for both SPs and clients. If the platform favors either party, then the other party may not have the incentive to participate. Therefore, the mission of the platform is to maximize *social welfare*, the sum of all the SP's and client's utilities, by matchings SPs with SCs. An SP is matched with an SC if the whole SC is placed in edge servers owned by the same SP. Since SPs and clients have their own interests, a set of matchings that maximizes the social welfare may not be the best choice of every SP or client. Explicitly, some SP or client may become a *blocking individual*, meaning that she or he can be better off by deviating from the matching result. Furthermore, a pair of SP and client that are not matched may become a *blocking pair* when both could be better off if they were matched to each other. Therefore, a crucial requirement on the solution is *stability*, which implies the exclusion of both blocking individuals and blocking pairs.

In this paper, we address the SCP problem in a multi-SP edge system. A VNF may demand a particular location to place (which is a locality constraint), yet every SC comes with a constraint on the aggregated latency including processing, propagation, and transmission delays. For a particular SC to be placed in a single SP, the objective of the SCP problem is to minimize the placement cost. The cost then becomes the minimal price for the placement. Since the prices may be different for different SPs and SCs, finding an optimal set of matchings between SPs and SCs that is stable yet maximizes social welfare is nontrivial.

We propose using the Deferred Acceptance (DA) algorithm [9] to generate a preliminary matching result. Because our definition of SP's preference on SCs are *substitutable*, the result is stable [10]. However, the result is also the worst stable matching for SPs. We thus use the T-algorithm [11] to make the resulting matching egalitarian.

Manuscript received January 1, 2021.
Manuscript revised January 1, 2021.
†The author is with Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, Taiwan.
DOI: 10.1587/transcom.E0.B.1

We conducted simulations to study the performance of the proposed approach and compared it with Boston Student Assignment Mechanism (BSAM) [12]. The result shows that the proposed approach can achieve higher social welfare yet fewer matching pairs than BSAM. Compared with pure DA, the proposed approach can improve SP's interests when placing many SCs.

The remainder of the paper is organized as follows. Section 2 reviews related works, present the system model, and formulates the problem. Section 3 defines preference functions for each client/SP and presents the proposed approach. Section 4 shows our simulation results. Section 5 concludes the paper.

2. Background and Problem Formulation

2.1 Related Work

Generally, VNF placement problem is to minimize the placement cost, either the cost by physical machines or the cost by traffic. The cost by physical machines comes in two major types, resource consumption [13] and computational cost [14]. Zanzi et al. [13] introduced multi-access edge computing (MEC) broker and modeled the problem as minimizing the overall capacity utilization over different MEC systems hosted in a 5G network. In [14], Benkacem et al. formulated the problem into two objectives, including minimizing the cost and maximizing the Quality of Experience (QoE) of virtual stream service in Content Delivery Network (CDN) slice. On the other hand, traffic cost is usually caused by system network traffic [15]. Carpio et al. [15] presented a way to improve load balancing in the NFV network by minimizing links utilization. Hyodo et al. [16] proposed a model that relaxes the visit order and no-loop constraints imposed by a logical network generated on an original physical network.

Apart from VNF placement, SCP also needs to address SFC routing. Some researches formulated routing cost based on the distance and allocated bandwidth between two consecutive VNFs [7] and aimed to minimize the traffic cost, including distance and allocated bandwidth of virtual links mapped to physical links, and operational cost represented by the number of active node between two consecutive time slot [7]. Luizelli et al. [17] tackled both propagation delay and processing delay by minimizing the number of VNFs mapped on the physical nodes. Cziva et al, [6] presented a dynamic placement scheduling solution for minimizing the expected end-to-end latency, considering the processing delay.

All studies mentioned above restricted the SCP problem to single SP. In reality, there may be more than one SPs with the same service coverage. This setting allows clients to select an SP that provides the lowest cost [8]. In this paper, we aim to maximize the sum of SP's profits and client's payoffs.

2.2 System Model

We assume F as the set of all possible VNFs and $S = \{s_1, s_2, ..., s_{|S|}\}$ as the set of all SCs to deploy. Each SC $s_k = (f_1^k, f_2^k, ..., f_{q_k}^k)$, where $f_i^k \in F$, is a sequence of q_k VNFs. The SC itself is associated with a latency constraint θ_k. Associated with each VNF f_i^k in s_k is the amount of requested computation resource, a set of areas allowed for deployment, and the requested bandwidth allocated to the logical link to the next VNF in s_k. We quantify computation resource as a number of computing resource blocks (CRBs) [18] and let γ_i^k be the number demanded by f_i^k. Define $G_k = (\gamma_1^k, \gamma_2^k, \cdots, \gamma_{q_k}^k)$. Assume that there are η areas in the system denoted by a set $A = \{a_1, a_2, ..., a_\eta\}$. Let $E = \{(u, v)|a_u, a_v \in A\}$ dnote the set of physical links between every two areas a_u and a_v. Each VNF f_i^k in s_k is allowed to be placed in one of the areas in area set $d_i^k \subseteq A$. The set of areas allowed by each VNF in s_k is denoted by $D_k = (d_1^k, d_2^k, ...d_{q_k}^k)$. A logical link $l_i^k = (f_i^k, f_{i+1}^k)$ is defined for each pair of two consecutive VNFs f_i^k and f_{i+1}^k in SC s_k. Each logic link is mapped onto a physical link. Let band_i^k be the amount of bandwidth requested by l_i^k. The set of bandwidth requested by each logical link in s_k is collectively denoted by $B_k = (\text{band}_1^k, \text{band}_2^k, \cdots, \text{band}_{q_k}^k)$.

We also assume a set of SPs $P = \{p_1, p_2, ...p_{|P|}\}$. Each SP $p_n \in P$ has accommodated NFV Management and Orchestration Architecture (NFV-MANO) and their own edge servers [19]. We use m_i^n to denote the edge server that p_n has placed in area $a_i \in A$ ($m_i^n = \emptyset$ if p_n does not place any edge server in a_i). The set of edge servers owned by p_n in all areas can then be captured by $M^n = (m_1^n, m_2^n, ..., m_\eta^n)$. To ease the deployment and management process, each SP p_n slices its resource and predetermines its quota $q(p_n)$, the maximum number of SCs admitted by p_n, and equally divides its computing resource into $q(p_n)$ blocks. Let c_i^n be the number of CRBs available in server m_i^n. The amount of CRBs allocated to each SC in m_i^n is then $c_i^n/q(p_n)$. All variables and notations used in this paper are summarized in Table 1.

For each SC $s_k \in S$, the associated client will broadcast an inquiry $r_k = (s_k, G_k, D_k, B_k)$ to all SPs asking for possible placement. If an SP p_n is able to place s_k, it will send an ask price to the client. The client then selects one SP (possibly the one with the lowest ask price) to send a placement request with a bid price. If an SP receives more requests than it can accept, it reject some requests (possibly those with low profits). The clients with requests rejected may then turn to other SPs or raise their bid prices and resubmit their requests. SP p_n and the client s_k may need several rounds of negotiations to reach their final price b_k^n.

2.3 Problem Formulation

Each SC s_k is associated with a budget ψ_k^n, the maximum price that the associated client is willing to pay to SP p_n for the placement of s_k. The budgets are differential because

Table 1: Summary of Notations

Overall	
P	Set of all SPs
A	Set of all areas
M	Set of all edge servers
E	Set of all physical links
S	Set of all SCs to deploy
F	Set of all possible VNFs
For service providers	
$p_n \in P$	The n-th SP in P
m_i^n	The edge server in area $a_i \in A$ owned by p_n
M^n	The set of all edge servers owned by p_n
$q(p_n)$	The maximum number of SCs admitted by p_n
c_i^n	The number of available CRBs in m_i^n
C_k^n	The minimal price asked by p_n to place SC $s_k \in S$
δ_j^n	Whether p_n owns an edge server in area $a_j \in A$
λ_j^n	The computation rate of m_j^n
$c_{n,u,v}^{\text{band}}$	The link bandwidth of $(u, v) \in E$ owned by p_n
$h_{u,v}$	The distance between two areas a_u and a_v
α_j^n	The unit cost of CRB in m_j^n
For service chains	
$s_k \in S$	The k-th SC in S
q_k	The length of s_k
θ_k	The latency constraint of s_k
ψ_k^n	The maximum payment to $p_n \in P$ for placing s_k
b_k^n	The final payment to $p_n \in P$ for placing s_k
L_k	The set of logical links of s_k
For VNF	
$f_i^k \in F$	The i-th VNF in SC s_k
γ_i^k	The number of CRBs demanded by f_i^k
d_i^k	The set of areas where f_i^k could be placed
$l_i^k \in L_k$	The logical link between f_i^k and f_{i+1}^k
band_i^k	The bandwidth demand of l_i^k
w_i^k	The workload of f_i^k
t_i^k	The traffic load of l_i^k
Output variables	
$x_{i,j}^{k,n}$	Whether f_i^k is placed in m_j^n
$y_{u,v}^{n,k,i}$	Whether l_i^k maps to $(u, v) \in E$ owned by p_n
Derived variable	
z_k^n	Whether s_k is served by p_n

different SPs may provide different levels of quality of service (QoS). On the other hand, each SP p_n has a minimum selling price C_k^n for the placement of s_k. The value of C_k^n depends on p_n's cost to place s_k. The matching between s_k and p_n is possible only if the final selling price b_k^n satisfies $C_k^n \le b_k^n \le \psi_k^n$. For this price, the client's utility is $\psi_k^n - b_k^n$ whereas p_n's utility is $b_k^n - C_k^n$.

Let $z_k^n \in \{1, 0\}$ indicate whether s_k is matched with p_n. The social welfare is defined as the sum of all SP's and client's utilities:

$$\begin{aligned}\sum_{p_n \in P} \sum_{s_k \in S} \left(z_k^n \left(\left(\psi_k^n - b_k^n\right) + \left(b_k^n - C_k^n\right)\right)\right) \\ = \sum_{p_n \in P} \sum_{s_k \in S} \left(z_k^n \cdot \left(\psi_k^n - C_k^n\right)\right), \qquad (1)\end{aligned}$$

Note that the social welfare has nothing to do with the final selling prices. Our objective is to maximize (1) subject to several constraints. The first few constraints concern **VNF placement**. First, each VNF in any SC is placed in at most one edge server. Let $x_{i,j}^{k,n} \in \{1, 0\}$ indicates whether VNF f_i^k is placed in edge server m_j^n. This constraint can be expressed as

$$\sum_{p_n \in P} \sum_{m_j^n \in M^n} x_{i,j}^{k,n} \le 1, \ \forall f_i^k \in s_k, \ \forall s_k \in S. \qquad (2)$$

Second, s_k is matched with p_n only if every VNF in s_k is placed in an edge server owned by p_n. Together with (2) we have

$$q_k \cdot z_k^n = \sum_{f_i^k \in s_k} \sum_{m_j^n \in M^n} x_{i,j}^{k,n}, \ \forall p_n \in P, \ \forall s_k \in S. \qquad (3)$$

A VNF in s_k can be placed in area a_j only if p_n has placed an edge server there:

$$x_{i,j}^{k,n} \le \delta_j^n, \ \forall m_j^n \in M^n, \ \forall p_n \in P, \ \forall f_i^k \in s_k, \ \forall s_k \in S, \quad (4)$$

where $\delta_j^n = 0$ if $m_j^n = \emptyset$ and $\delta_j^n = 1$ otherwise. Also, an SC's aggregated CRB requirement on any edge server cannot exceed the capacity allocated to the SC. In other words,

$$\sum_{f_i^k \in s_k} \left(\gamma_i^k \cdot x_{i,j}^{k,n}\right) \le \frac{c_j^n}{q(p_n)}, \forall s_k \in S, \forall p_n \in P, \forall m_j^n \in M^n. \qquad (5)$$

Yet another constraint is that the total number of SCs placed in any SP p_n cannot exceed p_n's quota:

$$\sum_{s_k \in S} z_k^n \le q(p_n), \forall p_n \in P. \qquad (6)$$

The next few constraints concern **SFC routing**. Let $y_{u,v}^{n,k,i} \in \{1, 0\}$ indicate whether the physical link (u, v) owned by p_n is allocated to logical link l_i^k. Any such allocation is allowed only if s_k is matched with p_n. That is,

$$y_{u,v}^{n,k,i} \le z_k^n,\ \forall l_i^k \in L_k,\ \forall (u,v) \in E,\ \forall p_n \in P,\ \forall s_k \in S. \tag{7}$$

Furthermore, the aggregated bandwidth requirement on a physical link $(u,v) \in E$ cannot exceed the bandwidth capacity. Let $c_{n,u,v}^{\text{band}}$ be the bandwidth capacity of link $(u,v) \in E$ owned by p_n, then this constraint is

$$\sum_{s_k \in S} \sum_{l_i^k \in L_k} \text{band}_i^k \cdot y_{u,v}^{n,k,i} \le c_{n,u,v}^{\text{band}},\ \forall (u,v) \in E,\ \forall p_n \in P. \tag{8}$$

To match s_k with p_n, we also must allocate an physical link to each logical link l_i^k in s_k. Eq. (9) ensures that the physical link allocated to l_i^k starts from the edge server where f_i^k is placed:

$$\sum_{m_v^n \in M^n \setminus \{m_u^n\}} y_{u,v}^{n,k,i} \ge x_{i,u}^{k,n},\ \forall f_i^k \in s_k,\ \forall m_u^n \in M^n,\ \forall p_n \in P. \tag{9}$$

Furthermore, (10) asserts that the logical link l_i^k from f_i^k to f_{i+1}^k is allocated a physical link from the edge server where f_i^k is placed to that where f_{i+1}^k is placed.

$$\sum_{m_v^n \in M^n} \left(y_{u,v}^{n,k,i} - y_{v,u}^{n,k,i}\right) = x_{i,u}^{k,n} - x_{i+1,u}^{k,n}, \forall f_i^k \in s_k,\ \forall (u,v) \in E. \tag{10}$$

Note that $x_{i,u}^{k,n} = x_{i+1,u}^{k,n}$ if f_i^k and f_{i+1}^k are placed in the same edge server m_u^n.

The latency constraint shown in (11) demands that s_k can match with p_n only if the total service latency does not exceed θ_k.

$$z_k^n \left(L_{\text{proc}}(k,n) + L_{\text{trans}}(k,n) + L_{\text{prop}}(k,n)\right) \le \theta_k,\ \forall s_k \in S, \tag{11}$$

where $L_{\text{proc}}(k,n)$, $L_{\text{trans}}(k,n)$, and $L_{\text{prop}}(k,n)$ denote the total processing delay, transmission delay, and propagation delay, respectively, when s_k is matched with p_n. Processing delay is the time it takes for an edge server to process the work load generated by a VNF instance hosted by it. Assuming computation capacity λ_j^n of edge server m_j^n and work load w_i^k of VNF f_i^k, the total processing delay of $s_k \in S$ matched with p_n can be estimated as

$$L_{\text{proc}}(k,n) = \sum_{f_i^k \in s_k} \sum_{m_j^n \in M^n} \frac{w_i^k}{\lambda_j^n} x_{i,j}^{k,n}. \tag{12}$$

Transmission delay is the amount of time required to push the traffic of logical link l_i^k into the allocated physical link. Assuming a traffic load t_i^k of link l_i^k and the bandwidth capacity $c_{n,u,v}^{\text{band}}$ of physical link (u,v) owned by SP p_n, the total transmission delay of $s_k \in S$ matched with s_p can be estimated as

$$L_{\text{trans}}(k,n) = \sum_{f_i^k \in s_k} \sum_{(u,v) \in E} \frac{t_i^k}{c_{n,u,v}^{\text{band}}} y_{u,v}^{n,k,i}. \tag{13}$$

Propagation delay is the time it takes to transmit some signal from the source to the destination. Assuming a signal speed as the speed of light c and letting $h_{u,v}$ be the physical distance between two areas a_u and a_v, we can formulate the propagation delay as

$$L_{\text{prop}}(k,n) = \sum_{f_i^k \in s_k} \sum_{(u,v) \in E} \frac{h_{u,v}}{c} y_{u,v}^{n,k,i},\ \forall s_k \in S. \tag{14}$$

3. Proposed Mechanism

The first step of the solution is to find out the minimal selling price C_k^n for each $s_k \in S$ and $p_n \in P$. The value of C_k^n is at least the cost of matching s_k with p_n with VNF placement decisions $\{x_{i,j}^{k,n} \mid 1 \le i \le q_k, 1 \le j \le \eta\}$. Let α_j^n be the cost per CRB in edge server $m_j^n \in M^n$, we have

$$C_k^n = \min_{x_{i,j}^{k,n}} \sum_{f_i^k \in s_k} \sum_{m_j^n \in M^n} \left(x_{i,j}^{k,n} \cdot \alpha_j^n \cdot \gamma_i^k\right) \tag{15}$$

subject to (2) to (11). It is an integer programming problem, which be solved by a general problem solver (e.g., Gurobi).

3.1 Individual's Preference

For the framework of matching, we need to define the *preference* of each each participant (either a client or an SP). A preference is a total order on the sets of all the possible matches for the participant. For SP's preference, we define a function $\phi_n(S')$ for SP p_n to evaluate a possible matching with a set of SCs $S' \subseteq S$. Because every SP prefers a set of SCs that brings in the maximal potential profit, we have

$$\phi_n(S') = \sum_{s_k \in S'} \left(\psi_k^n - C_k^n\right). \tag{16}$$

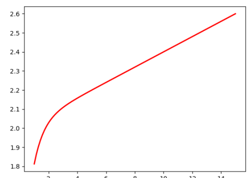
Since SP has limited resource capacity with limited service coverage, it is not necessary that $\phi_n(S') < \phi_n(S'')$ whenever $S' \subset S''$.

For clients, the client requesting SC s_k surely prefers an SP p_n to all other SPs if p_n's minimal selling price, C_k^n, is the lowest among all others. However, the client should also consider possible resource surplus provided by each SP. An SC may receive non-zero resource surplus because SPs generally do not allocate the exact amount of computing resource requested by the SC (recall that each SP p_n partitions its CRBs into $q(p_n)$ equal blocks). When an VNF f_i^k is hosted by an edge server m_j^n, the resource surplus is

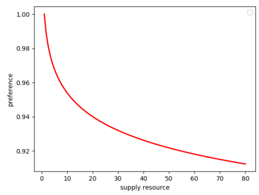
$$\rho_{i,j}^{k,n} = c_j^n / q(p_n) - \gamma_i^k. \tag{17}$$

If an SC has more resource surplus, the SC is more resilient against sudden spurt of resource demand.

We consider two preference functions for clients which obey the law of diminishing return [20]. The law states that the additions of resource surplus yield progressively smaller, or diminishing, increases in benefits after the amount of resource surplus reaches some point. It is considered a



(a)



(b)

Fig. 1: (a) The first preference function and (b) The second preference function.

general principle in Economics. The first preference function is defined as

$$\phi_k(p_n) = \sum_{f_i^k \in s_k} \left(\frac{c_j^n / q(p_n)}{C_k^n} \cdot \frac{\gamma_i^k}{\sum_{f_j^k \in s_k} \gamma_j^k} + \frac{\rho_{i,j}^{k,n} \cdot \int_{x=0}^{\rho_{i,j}^{k,n}} \lambda e^{-\lambda x} dx \cdot}{C_k^n} \cdot \frac{\rho_{i,j}^{k,n}}{\gamma_i^k} \right) \tag{18}$$

The second preference function is defined below:

$$\phi_k(p_n) = \sum_{f_i^k \in s_k} \frac{\gamma_i^k - \ln\left(\frac{c_j^n}{q(p_n)} - \gamma_i^k\right)}{C_k^n}. \tag{19}$$

With this function, the highest value occurs when the amount of resource allocated to s_k is exactly the same it demands. Fig. 1a and Fig. 1b show how the first and second preference functions, respectively, grow with increasing resource surplus.

With the definitions of preference functions, we can now define *preference relations* $\succ_{s_k}$ on P for every $s_k \in S$ and $\succ_{p_n}$ on 2^S for every $p_n \in P$. Formally, for each $s_k \in S$, $p_n \succ_{s_k} p_r$ (meaning that s_k prefers p_n to p_r) if and only if $\phi_k(p_n) > \phi_k(p_r)$. Similarly, for each $p_n \in P$, $S' \succ_{p_n} S''$ (p_n prefers S' to S'', where S' and S'' are two subsets of S) if and only if $\phi_n(S') > \phi_n(S'')$. Furethermore, we define $Ch(\mathcal{S}, \succ_{p_n}) = \{S' \subseteq \mathcal{S} \mid \nexists S'' \subseteq \mathcal{S}, S'' \succ_{p_n} S'\}$ for each $p_n \in P$, $\mathcal{S} \subseteq S$, and $Ch(\mathcal{P}, \succ_{s_k}) = \{p' \in \mathcal{P} \mid \nexists p'' \in \mathcal{P}, p'' \succ_{s_k} p'\}$ for each $s_k \in S, \mathcal{P} \subseteq P$.

3.2 Proposed Mechanism

We propose using the well-known DA [9] (Algorithm 1) first to generate a preliminary matching result for SCs and SPs. The mission of Algorithm 1 is to define two matching functions. One is $\mu_S : S \to P \cup \{\emptyset\}$, which is a mapping such that, for all $s_k \in S$, $\mu_S(s_k) = p_n$ if s_k is matched with p_n and $\mu_S(s_k) = \emptyset$ otherwise. The other is $\mu_P : P \to 2^S$, which is another mapping such that, for all $p_n \in P$, $\mu_P(p_n) = S'$ if p_n is matched with $S' \subseteq S$ and $\mu_P(p_n) = \emptyset$ otherwise. It is not difficult to see that Algorithm 1 ensures $\mu_S(s_k) = \{p_n\}$ if and only if $s_k \in \mu_P(p_n)$. For that property we call $\mu = \{\mu_S, \mu_P\}$ a *prematching*.

Algorithm 1 DA algorithm

Require: S; P
$\mu_P(p_n) \leftarrow \emptyset, \forall p_n \in P$ ▷ initialize p_n's matching result
$\mu_S(s_k) \leftarrow \emptyset, \forall s_k \in S$ ▷ initialize s_k's matching result
$\forall s_k \in S$: $s_k^{\text{list}} \leftarrow \{p_n \mid C_k^n \leq \psi_k^n\}$
$S_{to\_match} \leftarrow \{s_k \mid s_k \in S, s_k^{\text{list}} \neq \emptyset\}$
while $S_{to\_match} \neq \phi$ **do**
  $R_n \leftarrow \emptyset, \forall p_n \in P$ ▷ R_n keeps all requests to p_n
  for all $s_k \in S_{to\_match}$ **do**
    $p_n \leftarrow \arg\max_{p \in s_k^{\text{list}}} \phi_k(p)$ ▷ most preferred
    $R_n \leftarrow R_n \cup \{s_k\}$ ▷ new request to p_n
    $s_k^{\text{list}} \leftarrow s_k^{\text{list}} \setminus \{p_n\}$ ▷ no revisiting p_n
    if $s_k^{\text{list}} = \emptyset$ **then**
      $S_{to\_match} \leftarrow S_{to\_match} \setminus \{s_k\}$
    end if
  end for
  for all $p_n \in P$ such that $R_n \neq \emptyset$ **do**
    $A \leftarrow Ch(R_n \cup \mu_P(p_n), \succ_{p_n})$ ▷ all accepted requests
    $\mu_P(p_n) = A$
    $J \leftarrow (R_n \cup \mu_P(p_n)) \setminus A$ ▷ all rejected requests
    $S_{to\_match} \leftarrow S_{to\_match} \setminus A$
    for all $s_t \in A$ **do**
      $\mu_S(s_t) \leftarrow \{p_n\}$
    end for
    $S_{to\_match} \leftarrow S_{to\_match} \cup J$
    for all $s_t \in J$ **do**
      $\mu_S(s_t) \leftarrow \emptyset$
    end for
  end for
end while
return $(\{\mu_P(p)\}_{p \in P}, \{\mu_S(s)\}_{s \in S})$

The prematching is not necessarily stable. In our problem, SP's preference is *responsive* and thus substitutable because each SP has a fixed quota and (16). For that property the prematching is stable. Although the result is optimal for SCs, it is also the worst stable matching for SPs.

To make the matching egalitarian, we use T-algorithm [11] to find another matching from the prematching $\mu = \{\mu_S, \mu_P\}$. It attempts identifying two groups of sets from μ. The first $U(p_n, \mu_S) = \{s_k \in S \mid p_n \succeq_{s_k} \mu_S(s_k)\}$ is defined for each $p_n \in P$. Intuitively, an SC s_k is in $U(p_n, \mu_S)$ if either s_k is matched with p_n or s_k prefers p_n to the one matched with it. The second group of sets $V(s_k, \mu_P) = \{p_n \in P \mid \exists S' \subseteq S, s_k \in S' \cap Ch(\mu_P(p_n) \cup S', \succeq_{p_n})\}$ is defined for each $s_k \in S$. Intuitively, an SP p_n is in $V(s_k, \mu_P)$ if either p_n is matched with s_k or p_n would rather match with s_k than not match with s_k when considering the union of any subset of SCs that includes s_k and the set of SCs that is matched with p_n.

After identifying these two groups of sets, the T-algorithm iteratively updates the matching for each SC and SP. Explicitly, it updates $\mu_P(p_n)$ to $Ch(U(p_n, \mu_S), \succ_{p_n})$ for each $p_n \in P$ and updates $\mu_P(s_k)$ to $Ch(V(s_k, \mu_P), \succ_{s_k})$ for each $s_k \in S$. The iteration terminates when the above updating does not change any matching. Refer to Algorithm 2. It has been proved that the output of T-algorithm is stable provided that its input is stable [11].

Algorithm 2 T-algorithm

```
Require: S; P; {μ_P(p)}_{p∈P}; {μ_S(s)}_{s∈S}
μ' ← μ                                              ▷ initial prematching
repeat
    μ ← μ'
    U(p_n, μ_S) ← {s_k ∈ S | p_n ≥_{s_k} μ_S(s_k)} for all p_n ∈ P
    V(s_k, μ_P) ← {p_n ∈ P | ∃S' ⊆ S, s_k ∈ S' ∩ Ch(μ_P(p_n) ∪ S', ≥_{p_n})} for all s_k ∈ S
    μ'_P(p_n) ← Ch(U(p_n, μ_S), >_{p_n}) for each p_n ∈ P
    μ'_P(s_k) ← Ch(V(s_k, μ_P), >_{s_k}) for each s_k ∈ S
until μ' = μ
return μ'
```

3.3 Time Complexity Analysis

We shall now give bounds on the computational complexities of the proposed algorithms. In Algorithm 1, it is SCs that propose to SPs. From Lines 7 to 14, each SC s_k proposes to its most preferred SP p_n and remove it from s_k's preference list s_k^{list}. Since the size of s_k^{list} is at most $|P|$, each SC can make at most $|P|$ proposals. From this perspective, the time complexity of Algorithm 1 is $O(|S||P|)$. However, SCs could make their proposals simultaneously, making the algorithm executed in a round-by-round basis. If this is the case, the time complexity of Algorithm 1 is $O(|P|)$

The time complexity of T-algorithm relates to the total length of each SP's/SC's preference list. The length of each SC's preference list is at most $|P|$. Each SP p_n does not consider any subset of SCs with cardinality greater than $q(p_n)$. Therefore, the number of SC subsets $S' \subseteq S$ to consider in (16) is at most $\sum_{1\le i\le q(p_n)} \binom{|S|}{i}$. This implies that the length of p_n's preference list is $O(|S|^{q(p_n)})$. Echenique and Oviedo [11] have proved that the number of iterations in Algorithm 2 is less than

$$\sum_{s_k\in S}(\mathcal{L}_k - 1) + \sum_{p_n\in P}(\mathcal{L}_n - 1), \tag{20}$$

where $\mathcal{L}_k$ and $\mathcal{L}_n$ are the lengths of s_k's and p_n's preference lists, respectively. Therefore, the time complexity of Algorithm 2 is $O(|S||P| + |P||S|^{q(p_n)})$, which is $O(|P||S|^q)$ where $q = \max_{p_n\in P}\{q(p_n)\}$.

4. Numerical Results

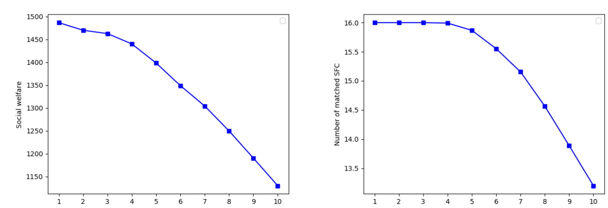
4.1 Simulation Settings

Because no settings or data of real trace were publicly available, we used simulations with synthetic parameters to investigate the performance of the proposed approaches. Though the results presented here do not reflect any real system in operation, we believe that the results still provide some insights into the problem under consideration.

Our simulations considered four SPs and η areas, where η ranged from 5 to 8. The total number of CRBs owned by each SP was uniformly distributed over $[600, 700]$ with default value 650. Each SP p_n deployed its edge serves in A_n out of η areas and equally allocated its CRBs to all its edge servers. There were 15 to 20 SCs (15 by default) in the simulations. We assumed 10 different types of VNFs. The lengths of SCs were randomly generated. The quota of each SP was fixed to 4. Parameter α_j^n in (15) was randomly selected from $[1, 1.5]$. The value of each ψ_k^n was uniformly distributed over $[210, 300]$.

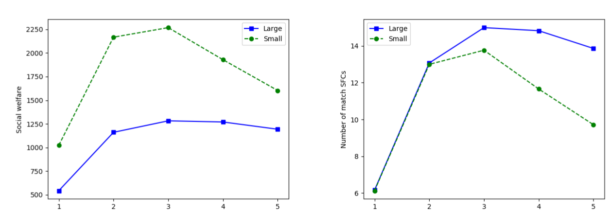
4.2 The Effect of The Length of SCs

We assumed 20 SCs, $\eta = A_n = 5$ for all p_n, and set q_k (the length of SC) by an exponential distribution. Each result is an average over 500 trials. Figs. 2a and 2b show how the social welfare and the number of matched SCs, respectively, changed when the mean of the distribution increased. Since the resource capacity of SPs was fixed, the increase in the resource demands (i.e., the mean number of VNFs requested) resulted in fewer deployed SCs and thus lower social welfare.



(a) (b)

Fig. 2: (a) The social welfare and (b) the number of matched SCs with increasing mean of $|s_k|$ (with 20 SCs)



(a) (b)

Fig. 3: (a) The social welfare and (b) the number of matched SCs versus the number of SP's service areas ('Large' has twice the resource capacity of 'Small')

4.3 The Effect of SP's Service Coverage

We fixed $\eta = 5$ and varied A_n, the number of p_n's service areas for each SP p_n. Since each SP equally allocated all its computing resource to all areas where the SP had deployed edge servers, the resource capacity of each edge server became smaller when the SP served more areas. Figs. 3a and 3b show the social welfare and the number of matched SCs, respectively, with increasing number of SP's service areas. Here each VNF is allowed to be deployed in one of three areas randomly selected from five areas. Both the number of matched SCs and the social welfare increase firstly. When an SP deployed edge servers in three out of five areas, for each VNF there is at least one area for the SP to deploy the VNF (by the pigeonhole principle). When an SP extended its service coverage to more areas, the number of matched SCs as well as the social welfare dropped because of relatively low resource capacity in each area. The setting of low resource capacity ('Small') had higher social welfare than the setting of high resource capacity ('Large') simply because the former had smaller expected resource surplus and thus lower resource cost.

We performed another set of experiments with $\eta = 8$ and $|d_i^k| = 5$ for all i and k. The values of A_n's were exponentially distributed with means ranged from 1 to 8. We used a normal distribution with $\mu = 3$ and varied σ^2 to generate q_k's (values truncated at 1 and 10). The result is shown in Fig. 4. Since $|d_i^k| = 5$ for all i and k, SP p_n can meet the locality constraint of any VNF when $A_n \geq 4$ (by the pigeonhole principle). Unlike 3b, where the number of matched SCs dropped when SPs extended their service coverage to excessive areas, the number of matched SCs in this setting did not decrease when the mean of A_n was larger than 4. The reason is that different SP may have different A_n value, thanks to the exponential distribution. Another factor that affects the result is the length of SC. A larger variance caused fewer matched SCs because a larger variance indicates a higher probability of a long SC, which demands more resource than a short SC.

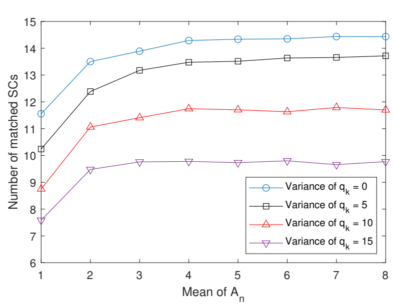
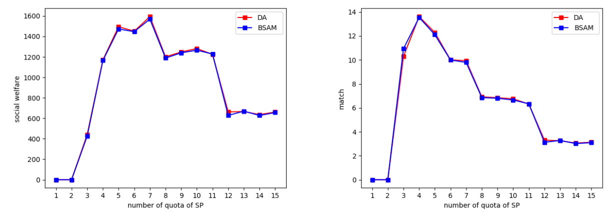


Fig. 4: The number of matched SCs with exponentially distributed A_n's ($\eta = 8$, $|d_i^k| = 5$ for all i and k)

4.4 The Effect of Different Quotas

Fig 5 shows the curves of the social welfare and the number of matched SCs versus the value of SP's quotas. When we increased the quota to be larger than four, the number of matched SCs decreased due to lower resource capacity in each area. However, the social welfare still increased because of lower expected placement cost due to lower resource surplus.



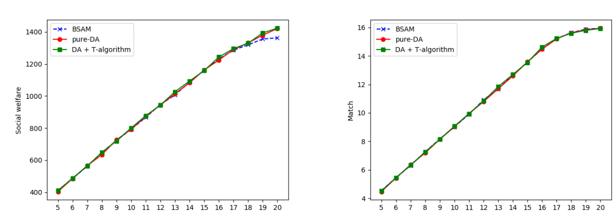
(a) (b)

Fig. 5: (a) The social welfare and (b) the number of matched SCs versus the value of SP's quotas

4.5 Comparison with BSAM and Pure DA

We compared the proposed approach with BSAM [12] and pure DA. We varied the number of SCs form 5 to 20. Figs. 6 and 7 show the results with the quota set to 4 and 3, respectively. The results indicate that BSAM is inferior to the other two counterparts concerning social welfare but superior to the others concerning the number of matched SCs. This is because SPs in BSAM do not reject an SC after accepting it. Therefore, SPs may be matched with less preferred SCs.

The proposed approach and pure DA do not differ significantly in both metrics except the social welfare tested with 20 SCs. For each SP p_n, we use p_n's preference function $\phi_n(S')$ to sort all possible subsets of SCs, S', in a non-increasing order. Fig. 8 shows the ordinal number of $\mu_P(p_n)$ in the sorted list for each SP p_n. We can see that $\mu_P(p_n)$ for each SP p_n has a slightly smaller ordinal number with the proposed approach (DA+T) than with pure DA. This confirms the effectiveness of the T-algorithm.

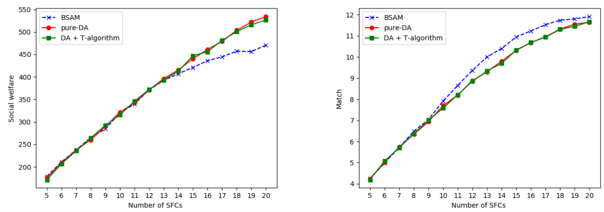


(a) (b)

Fig. 6: a) The social welfare and (b) the number of matched SCs versus the number of SCs ($q(p_n) = 4$)

5. Conclusions

We have studied SCP in multi-SC multi-SP edge system. We have formulated the social welfare maximization problem and proposed a two-sided matching mechanism based on DA and T-algorithm. The result is stable yet egalitarian. Simulation results show that social welfare may not be aligned with the number of matched SCs. The proposed approach outperforms BSAM in terms of social welfare but



(a) (b)

Fig. 7: a) The social welfare and (b) the number of matched SCs versus the number of SCs ($q(p_n) = 3$)

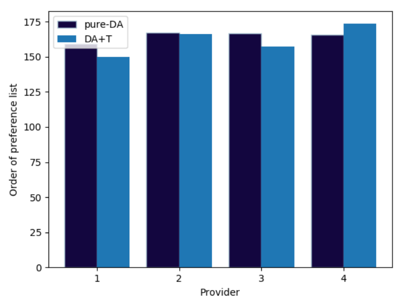


Fig. 8: The ordinal number of $\mu_P(p_n)$ for each SP p_n ($q(p_n) = 3$, $|S| = 20$)

not in terms of matched SCs.

In the future, we will add cloud data center into our system model. Cloud data center has plenty of computing resource which helps reducing the processing delay. However, its propagation delay is higher than edge systems. We will also consider an on-line approach to SCP, which handles SC placement requests in a one-by-one manner.

References

[1] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," IEEE Commun. Surveys Tuts., vol.18, no.1, pp.236–262, 2016.

[2] W. John, K. Pentikousis, G. Agapiou, E. Jacob, M. Kind, A. Manzalini, F. Risso, D. Staessens, R. Steinert, and C. Meirosu, "Research directions in network service chaining," IEEE SDN for Future Networks and Services, pp.1–7, 2013.

[3] A. Fischer, J.F. Botero, M.T. Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," IEEE Commun. Surveys Tuts., vol.15, no.4, pp.1888–1906, 2013.

[4] D. Li, P. Hong, K. Xue, and j. Pei, "Virtual network function placement considering resource optimization and SFC requests in cloud datacenter," IEEE Trans. Parallel Distrib. Syst., vol.29, no.7, pp.1664–1677, 2018.

[5] S. Song and J. Chung, "Sliced NFV service chaining in mobile edge clouds," Proc. 19th Asia-Pacific Netw. Operations and Manag. Symp., pp.292–294, 2017.

[6] R. Cziva, C. Anagnostopoulos, and D.P. Pezaros, "Dynamic, latency-optimal vNF placement at the network edge," Proc. IEEE INFOCOM, pp.693–701, 2018.

[7] C. Pham, N.H. Tran, S. Ren, W. Saad, and C.S. Hong, "Traffic-aware and energy-efficient vNF placement for service chaining: Joint sampling and matching approach," IEEE Trans. on Serv. Comput., vol.13, no.1, pp.172–185, 2020.

[8] D. Dietrich, A. Abujoda, A. Rizk, and P. Papadimitriou, "Multi-provider service chain embedding with Nestor," IEEE Trans. Netw. Service Manag., vol.14, no.1, pp.91–105, 2017.

[9] D. Gale and L.S. Shapley, "College admissions and the stability of marriage," Am. Math. Mon., vol.69, no.1, pp.9–15, 1962.

[10] A.S. Kelso and V.P. Crawford, "Job matching, coalition formation, and gross substitutes," Econometrica, vol.50, no.6, pp.1483–1504, 1982.

[11] F. Echenique and J. Oviedo, "Core many-to-one matchings by fixed-point methods," J. Econ. Theory, vol.115, no.2, pp.358–376, 2004.

[12] A. Abdulkadiroğlu and T. Sönmez, "School choice: A mechanism design approach," Am. Econ. Rev., vol.93, no.3, pp.729–747, June 2003.

[13] L. Zanzi, F. Giust, and V. Sciancalepore, "M2EC: A multi-tenant resource orchestration in multi-access edge computing systems," Proc. IEEE Wireless Commun. and Netw. Conf., 2018.

[14] I. Benkacem, T. Taleb, M. Bagaa, and H. Flinck, "Optimal VNFs placement in CDN slicing over multi-cloud environment," IEEE J. Sel. Areas Commun., vol.36, no.3, pp.616–627, 2018.

[15] F. Carpio, S. Dhahri, and A. Jukan, "VNF placement with replication for load balancing in NFV networks," Proc. IEEE Int. Conf. Commun., 2017.

[16] N. Hyodo, T. Sato, R. Shinkuma, and E. Oki, "Virtual network function placement for service chaining by relaxing visit order and non-loop constraints," IEEE Access, vol.7, pp.165399–165410, 2019.

[17] M.C. Luizelli, W.L. da Costa Cordeiro, L.S. Buriol, and L.P. Gaspary, "A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining," Comput. Commun., vol.102, pp.67–77, 2017.

[18] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F.R. Yu, and Z. Han, "Computing resource allocation in three-tier IoT Fog networks: A joint optimization approach combining Stackelberg game and matching," IEEE Internet Things J., vol.4, no.5, pp.1204–1215, Oct. 2017.

[19] V. Sciancalepore, F. Giust, K. Samdanis, and Z. Yousaf, "A double-tier MEC-NFV architecture: Design and optimisation," IEEE Conf. on Standards for Commun. and Netw., 2016.

[20] R.W. Shephard and R. Färe, "The law of diminishing returns," Production Theory, Berlin, Heidelberg, pp.287–318, Springer Berlin Heidelberg, 1974.