文章编号:1671-6833(2022)06-0022-08

一种收益最大化的服务功能链优化编排算法

黄 骅[1,2],江 俊[3],杨永康[2],何德峰[1],曹 斌[4]

(1.浙江工业大学 信息工程学院,浙江 杭州 310023;2.东方通信股份有限公司,浙江 杭州 310053;3.浙江树人大学 信息科技学院,浙江 杭州 310053;4.浙江工业大学 计算机科学与技术、软件学院,浙江 杭州 310023)

摘 要:针对数据中心内的服务功能链优化编排的问题,通过分析请求到达率、计算资源与处理延时之间的关系,以最大化部署收益为优化目标,构建了一种基于弹性资源分配的服务功能链优化编排模型,在此基础上提出一种收益最大化的服务功能链编排算法。该算法分为 2 个阶段,第 1 阶段,在满足处理时长的前提下优化服务功能链的资源消耗,在满足传输时延约束的前提下提高资源使用率,降低资源开销。第 2 阶段,面向虚拟网络功能部署及映射链路,基于 worst-fit 策略,交替采用自上而下和自下而上 2 种搜索策略,以提升部署效率,最大化计算资源和链路资源利用率。针对 pod 数为 4 和 6 两种场景,设计了仿真实验以验证启发式算法的性能。实验结果表明:相比现有算法,本文算法在部署收益、部署成功率和资源利用率 3 个指标方面均有一定提升,能够实现服务资源的优化配置,有效提升部署收益。

关键词:网络功能虚拟化;服务功能链;弹性分配;组合优化

中图分类号:TP393 **文献标志码**:A **doi**:10.13705/j.issn.1671-6833.2022.03.017

0 引言

网络功能虚拟化(network function virtualization, NFV)是实现 5G 服务化架构的关键技术之一。NFV 将各类网元软件化为虚拟网络功能(virtualized network function, VNF),使网络摆脱了对传统硬件设备的依赖,便于网络设备服务的更新升级,降低了运营成本,提升了业务部署的弹性。随着 NFV 应用研究的不断深入,一系列关键问题被提出并成为研究热点。

在基于 NFV 的网络中,业务功能主要由服务功能链[1-2](service function chain, SFC)承载。SFC 由一系列需要特定数量资源的 VNF 实例组成,每一个 VNF 分配的计算、存储、带宽等资源及部署的物理位置都会对端到端的 QoS 性能产生一定影响。因此需要设计一种最优部署方案以提升服务性能,这类问题被称为服务功能链编排(service function chain orchestration, SFCO)[3]。另一方面,随着 NFV 应用的不断拓展,企业或个人用户可以将各自的 IT 服务请求外包给运营商以获得更加高效且价格低廉的网络服务。一般情况下,运营商通过数据中心承载来自客户的服务请求。因此如何在满足要求的前提下,优化数据中心的资源利用率、降低部署成本是一个亟待解决的问题。

SFC 编排要求在满足各类约束条件的前提下最大化部署收益,而相关研究多将 VNF 的处理速率作为常量处理。实际应用场景中,VNF 分配的资源量与性能存在一定关系,现有研究表明,通过资源动态分配技术可以增强部署灵活性,实现编排结果的进一步优化[4-5]。

基于上述研究思路,本文聚焦于数据中心内的 SFC 编排问题,提出一种收益最大化的编排算法。结合资源弹性分配机制,在满足约束的前提下优化服务功能链的资源消耗,设计了启发式算法求解最优编排策略,并通过实验对算法的性能进行了评估。

1 相关工作

近年来,服务功能链编排问题受到了业界的广泛关注,取得了丰富的成果。SFC 编排属于 NP-Hard 问题[6],当服务功能链数量较多或物理

收稿日期:2022-04-11;**修订日期**:2022-06-23

基金项目:国家重点研发计划项目(2018YFB1402802);浙江省教育厅科学研究计划资助项目(Y202146607)

作者简介:黄骅(1983—),男,浙江杭州人,工程师,浙江工业大学博士、博士后,主要从事预测控制、服务计算方面的研究,E-mail:huanghua@eastcom.com。

网络规模较大时,很难在有限时间内找到最优解。因此,学者们引入启发式方法实现兼顾计算速度和优化效果的编排策略。

Li 等[7]考虑随时间变化的工作负载和基本资源消耗,将数据中心内部的 SFC 编排问题建模为整数线性规划求解。Li 等[8]提出了 NFV-RT 算法,在数据中心内给定端到端延迟约束的情况下最大化接受请求的数量。Qiao 等[9]考虑数据中心内的动态 SFC 编排问题,以最小化端到端时延为目标,结合遗传算法和模糊策略设计了启发式算法。Yu 等[10]考虑数据中心内部的 SFC 编排场景,以带宽资源最优化为目标建立了整数线性规划模型,并设计了两阶段算法求解该问题。Thai 等[11]提出一种两阶段算法求解数据中心内的 SFC 编排问题,第 1 阶段采用贪心策略挑选距离当前节点时延最小的 VNF 实例,第 2 阶段通过尝试其他可用的 VNF 实例来替换第 1 阶段中的解,并在允许条件下交换 SFC 中 VNF 的顺序以尝试搜索更优的编排方案。此外,一些学者将强化学习引入以解决 SFC 编排问题,例如 Liu 等[12]提出的基于强化学习的服务功能链映射方法。

目前关于 SFC 编排的研究大多将 VNF 的处理速率作为常量处理,然而在实际应用场景中,VNF 处理速率会受到资源分配、请求到达率等多种因素的影响。已有部分学者在模型中考虑 VNF 的动态特性,例如 Alleg 等[13]结合 VNF 的弹性资源分配机制,以最小化网络时延和能耗为目标建立了弹性资源分配模型(flexible resource allocation model, FRAM),动态调整 VNF 的资源分配,优化 SFC 传输性能,并通过求解整数线性规划得到资源分配及部署方案,该工作的不足之处在于未综合考虑请求到达率、处理时延与资源分配之间的关系以及基于整数规划的算法时间开销过大问题。为解决这一问题,本文将 FRAM 模型引入数据中心 SFC 编排场景中,分析请求到达率、计算资源与处理延时之间的关系,以最大化部署收益为优化目标,构建了基于弹性资源分配的编排模型,并设计了两阶段启发式算法求解优化问题。

2　系统模型

2.1　问题定义

在 NFV 架构中,网络服务主要由 SFC 承载,SFC 是指有序的 VNF 序列。图 1 为 5 个 VNF 组成的服务功能链,数据流从起始点开始依次经过防火墙、深度包检测、加密、网络监视和解密,最后到达终点。VNF 编排请求到达后,首先,需要选择最优节点并求解出最优路径,其次,以此为条件将 VNF 逐个实例化至服务器上,完成服务功能链的构建。

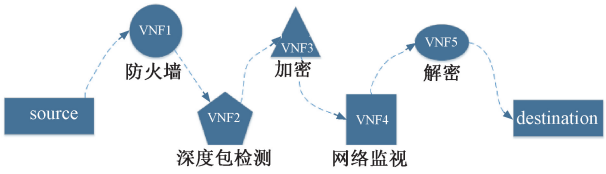

图 1　服务功能链示例

Figure 1　An example of service function chain

本文将数据中心网络建模为无向图 $G=(N,L)$,参数定义见表 1,其中 $N=\{n_1,n_2,\cdots,n_{|N|}\}$。在服务器内部署 VNF 需要占用一定的 CPU 和内存,为处理方便,将 CPU 资源和内存资源统一处理为计算资源。

2.2　数据中心网络拓扑

本文研究基于胖树型[14](fat tree)网络拓扑,该拓扑自上而下分为边缘层、汇聚层、核心层以及服务器层。对于 k 叉 fat tree 来说,共包含 k 个 pod,每个 pod 内的边缘交换机及聚合交换机数量均为 $k/2$。核心交换机的数量为 $(k/2)^2$,汇聚层和边缘层交换机的数量均为 $k^2/2$,服务器总量为 $k^3/4$。本文假设所有的 VNF 只能部署在服务器层,令数据中心内的节点集合为 N,交换机集合为 N_{sw},服务器集合为 N_s。

2.3　动态资源分配模型

本节研究资源分配对 VNF 性能和端到端时延的影响。采用 $M/M/1$ 排队模型模拟 VNF 服务,由于数据中心内部可以忽略传输耗时,所以只需要考虑处理耗时和排队耗时。由排队论[2]可知,d_i^j 和 κ_i、μ_i^j 之间存在如下关系:

$$d_i^j=\frac{1}{\mu_i^j-\kappa_i}。\tag{1}$$

在实际应用场景中,服务率 μ_i^j 与计算资源存在一定关系。本文采用弹性资源分配方式[13],假设服务率 μ_i^j 与资源数量 Φ_i^j 之间存在如下分段线性关系:

$$\mu_i^j=f(\Phi_i^j)=\begin{cases}a_i^j\Phi_i^j+b_i^j, \min\Phi_i^j\leqslant\Phi_i^j\leqslant\max\Phi_i^j;\\ \max\mu_i^j, \Phi_i^j\geqslant\max\Phi_i^j。\end{cases}\tag{2}$$

则参数 a_i^j 与 b_i^j 分别为

$$a_i^j=\frac{\max\mu_i^j-\min\mu_i^j}{\max\Phi_i^j-\min\Phi_i^j};\tag{3}$$

表1 各参数定义

Table 1 Definition of parameters

分类	参数	描述
网络	N	数据中心网络节点集合
	N_{sw}	数据中心交换机集合
	N_s	数据中心服务器集合
	E	网络链路集合
	$l^{m,n}, l^{m,n} \in E$	节点 m、n 之间的链路
	$res_n, n \in N$	虚拟机 n 的可用计算资源
	$res_{m,n}^{bw}, m, n \in N$	链路 $l^{m,n}$ 的最大可用带宽
SFC	S	服务功能链集合,$s_i \in S$ 为集合 S 中的任一服务链
	$s_i = \{s_i^1, s_i^2, s_i^j\}, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	服务链 s_i 的 VNF 集合
	s_i^j	s_i 的第 j 个 VNF
	D_i	s_i 的时延阈值
	$l_i^{j,j+1}$	s_i^j 和 s_i^{j+1} 之间的逻辑链路
	$\kappa_i, i \in \lvert S\rvert$	s_i 的服务到达率(请求数/s)
	$\mu_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	s_i^j 的服务率(请求数/s)
	$d_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	s_i^j 的平均处理时间
	$\Phi_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	s_i^j 分配的计算资源数
	$\rho_i, i \in \lvert S\rvert$	s_i 的带宽资源需求
	$r_i, i \in \lvert S\rvert$	s_i 的部署收益
	$\min \Phi_i^j, \max \Phi_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	s_i^j 资源分配的最小、最大值
	$\min \mu_i^j, \max \mu_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	s_i^j 的最小、最大服务率
决策变量	$x_n^{i,j}$	是否将 s_i 的第 j 个 VNF 部署于服务器 n 上,若部署则取1,反之取0
	$y_{m,n}^{i,u,u+1}$	逻辑链路 $l_i^{u,u+1}$ 是否经过 $l^{m,n}$,若经过取1,反之取0
	$z_i, i \in \lvert S\rvert$	s_i 是否成功部署,成功部署为1,反之为0
	$\lambda_i^j, i \in \lvert S\rvert, j \in \lvert s_i\rvert$	资源分配基数
最优指标权重系数	σ	节点映射开销系数
	μ	链路映射开销系数

$$b_i^j = \frac{\max \mu_i^j \cdot \min \Phi_i^j - \min \mu_i^j \cdot \max \Phi_i^j}{\min \Phi_i^j - \max \Phi_i^j}。\quad (4)$$

为简化描述,记 $\Phi_i^j = \lambda_i^j \min \Phi_i^j$,$d_i^j$ 的取值满足以下约束:

$$d_i^j = \frac{1}{\mu_i^j - \kappa_i} = \frac{1}{a_i^j \Phi_i^j + b_i^j - \kappa_i} = \frac{1}{a_i^j \lambda_i^j \min \Phi_i^j + b_i^j - \kappa_i};\quad (5)$$

$$1 \leqslant \lambda_i^j \leqslant \max \Phi_i^j / \min \Phi_i^j;\quad (6)$$

$$a_i^j \lambda_i^j \min \Phi_i^j + b_i^j > \kappa_i。\quad (7)$$

由式(2)可知,随着分配资源的增加,VNF 的服务率呈线性上升。若 $\Phi_i^j < \min \Phi_i^j$,则无法部署 s_i^j;若 $\Phi_i^j = \max \Phi_i^j$,s_i^j 的服务率达到最大值,即处理能力达到上限。

式(1)~(7)在 FRAM 模型的基础上进一步分析了处理延时、请求到达率和资源分配之间的关系。其中,式(6)为资源分配基数的取值范围,式(7)表明服务率必须大于到达率,反之会导致请求积压。

2.4 系统约束

首先,当 VNF 部署时,需要占用一定的计算资源且不能超过该服务器所能提供的资源上限,因此有以下约束:

$$\sum_i^{\lvert S\rvert} \sum_j^{\lvert s_i\rvert} x_n^{i,j} \cdot \min \Phi_i^j \cdot \lambda_i^j \leqslant res_n, n \in N。\quad (8)$$

对于数据中心内的任意链路 $l^{m,n}$,经过该链路所有 SFC 占用的带宽之和不能超过该链路的可用带宽,因此存在以下约束:

$$\sum_i^{\lvert S\rvert} \sum_u^{\lvert s_i\rvert - 1} \rho_i \cdot y_{m,n}^{i,u,u+1} \leqslant res_{m,n}^{bw}, n, m \in N。\quad (9)$$

此外,由于任意 VNF 只能映射至1台服务器,因此以下约束成立:

$$\sum_n^{\lvert N\rvert} x_n^{i,j} = 1, i \in \lvert S\rvert, j \in \lvert s_i\rvert。\quad (10)$$

其次,假设有虚拟链路 $l_i^{u,u+1}$ 在传输过程路由唯一,即对于任意 $l_i^{u,u+1}$ 存在以下约束:

$$\sum_{m,n}^{N} y_{m,n}^{i,u,u+1} = \begin{cases} 1,\text{存在} n \in \varphi(m) \text{ 满足 } y_{m,n}^{i,u,u+1} = 1; \\ 0,\text{其他}。 \end{cases} \quad (11)$$

最后,需要满足 s_i 的端到端时延约束要求。假定任意服务链 s_i 的端到端时延由传输时延、处理时延和排队时延 3 部分组成。由于 SFC 部署于数据中心内部,此时可以忽略传输时延约束,处理时延和排队时延之和为 $\sum_{j}^{|s_i|} \frac{1}{a_i^j \lambda_i^j \min \Phi_i^j + b_i^j - \kappa_i}$ 且小于等于该 s_i 的时延阈值,即以下约束成立:

$$\sum_{j}^{|s_i|} \frac{1}{a_i^j \lambda_i^j \min \Phi_i^j + b_i^j - \kappa_i} \leqslant D_i, i \in N。 \quad (12)$$

2.5 费用模型

部署收益 *Prof* 为 SFC 的总营收减去部署开销。总营收 R 为各 SFC 的营收之和,即 $R = \sum_{i}^{|S|} r_i z_i$。部署开销包括节点映射开销和链路映射开销,前者是由于实例化占用虚机资源造成,后者是占用链路带宽造成。记节点映射开销为 c_{node},链路映射开销为 c_{link},计算式为

$$c_{\text{node}} = \sigma \sum_{i}^{|S|} \left(r_i \sum_{j}^{|s_i|} \Phi_i^j \right); \quad (13)$$

$$c_{\text{link}} = \mu \sum_{i} r_i \left(\sum_{j}^{|s_i|-1} \left(\rho_i \sum_{m} \sum_{n} y_{m,n}^{i,j,j+1} \right) \right)。 \quad (14)$$

则部署收益 *Prof* 为

$$Prof = R - c_{\text{node}} - c_{\text{link}} = \sum_{i}^{|S|} r_i z_i - \sigma \sum_{i}^{|S|} \left(r_i \sum_{j}^{|s_i|} \Phi_i^j \right) - \mu \sum_{i} r_i \left(\sum_{j}^{|s_i|-1} \left(\rho_i \cdot \sum_{m} \sum_{n} y_{m,n}^{i,j,j+1} \right) \right)。 \quad (15)$$

2.6 问题定义

基于以上研究,SFC 优化编排问题可表述为

$$\max \left(\sum_{i}^{|S|} r_i z_i - \sigma \sum_{i}^{|S|} \left(r_i \sum_{j}^{|s_i|} \Phi_i^j \right) - \mu \sum_{i} r_i \left(\sum_{j}^{|s_i|-1} \left(\rho_i \sum_{m} \sum_{n} y_{m,n}^{i,j,j+1} \right) \right) \right)。 \quad (16)$$

其中,式(16)的优化目标是最大化部署收益,即在满足约束(式(6)~(12))的前提下最大化营收与部署开销之差。

3 算法描述

SFC 编排为 NP-hard 问题[6],本文引入弹性资源分配机制后进一步增加了其复杂性。为了有效解决该问题,提出一种启发式策略,通过对资源动态调整,在时延约束允许范围内尽可能降低资源耗费,增加部署成功率,提升部署收益。

3.1 动态资源调整策略

式(5)~(7)给出了 SFC 处理耗时与计算资源分配之间的关系。结合式(16)可知,提高收益的一种可行的策略是在约束允许的范围内尽可能降低资源消耗。本文采用动态资源调整策略,即在满足取值范围和时延约束的前提下尽可能增大计算资源使用率,对任意 SFC,在满足式(6)、(7)、(12)的条件下,求解以下最优化问题:

$$\min \sum_{j}^{|F_i|} a_i^j \lambda_i^j \cdot \min \Phi_i^j。 \quad (17)$$

3.2 SFC 部署算法

SFC 部署算法的核心思路:根据 worst-fit 策略挑选剩余带宽资源最多的链路,并且使同一个 SFC 中的所有 VNF 尽可能部署在一台服务器上,在实现负载均衡的同时降低链路部署花费[15]。

令 $Adj_{\text{down}}(n)$ 代表所有与网络节点 n 相邻且在网络拓扑中处于更低一层的节点集合。即当 $m \in Adj_{\text{down}}(n)$ 时,m 与 n 相邻且位于 n 的下一层。同理,可定义 $Adj_{\text{up}}(n)$ 为所有与节点 n 相邻且在网络拓扑中处于更高一层的节点集合。所有的节点 n 需要维护变量 $Cap(n) = res_{\text{reminder}}(n)$,当 $n \in N_s$ 时,$res_{\text{reminder}}(n)$ 为节点 n 的剩余可用资源;当 $n \in N_{sw}$ 时,$res_{\text{reminder}}(n)$ 为集合 $Adj_{\text{down}}(n)$ 中所有节点 $res_{\text{reminder}}(n)$ 的最大值。

部署 s_i 时,首先,在根节点调用向下搜索策略(searchDown),寻找服务器以部署 s_i 的第 1 个 VNF,之后,对于 s_i 中每个待部署的 VNF,尽可能把它部署在已承载了前一个 VNF 的服务器处,如果该服务器的剩余资源无法实例化 VNF,则调用向上搜索策略(searchUp),从服务器层出发向更高层寻找一个合适的交换机节点;其次,从交换机节点处调用向下搜索策略寻找一个满足部署条件的服务器节点;最后,返回 s_i 的部署方案。searchDown 和 searchUp 算法流程如下所示。

算法 1 寻找合适的服务器节点(searchDown)。

输入:待部署的 VNF 的 s_i^j 以及当前所在节点 n;

输出:部署方案 *nodeList* 及路由方案 *pathList*。

① 初始化输入参数,令 $nodeList = \varnothing$,$pathList = \varnothing$,$unavailableList = \varnothing$;

② if $n \notin N_s$, do

③ for all $m \in Adj_{\text{down}}(n)$

④ 寻找节点满足:链路 $l^{m,n}$ 的带宽大于等于 ρ_i;部署 s_i^j 后 $l^{m,n}$ 剩余的可用带宽最大;$Cap(m)$ 大于等于 s_i^j 的资源需求;m 不属于集合 *unavailableList*。

⑤ if 存在满足上述条件的节点 m
⑥ 令 $n=m$,将 m 添加至 *nodeList*,将 $l^{m,n}$ 添加至 *pathList*
⑦ else
⑧ if *nodeList* = ∅ or *pathList* = ∅
⑨ s_i^j 无法部署
⑩ return;
⑪ else
⑫ 将 m 添加至 *unavailableList*,令 $n=tail(nodeList)$,删除 $tail(nodeList)$和 $tail(pathList)$
end if
⑬ end if
⑭ end for
⑮ end if
⑯ return *nodeList*, *pathList*

当部署第 1 个 VNF 或当前服务器无法满足部署 VNF 的资源需求时,调用 searchDown 算法。该算法的作用是从网络拓扑较高层开始搜索,逐层挑选剩余带宽最多的链路。同时为了满足资源约束,挑选链路时需要确保另一端节点的 *Cap* 值大于等于 s_i^j 的部署需求。如果不存在满足要求的节点,则 s_i^j 无法部署。反之则重复搜索直至找到满足条件的服务器节点,输出 *nodeList* 和 *pathList*。

算法 2 寻找合适的交换机节点(searchUp)。

输入:待部署的 s_i^j 以及当前所在节点 n;

输出:合适的交换机节点 *nodeList* 及路由方案 *pathList*。

① 初始化输入参数,令 *nodeList* = ∅, *pathList* = ∅, *unavailableList* = ∅;
② for all $m \in Adj_{up}(n)$,寻找节点 m 满足:$l^{m,n}$ 的剩余带宽大于等于 ρ_i;$l^{m,n}$ 部署 s_i^j 后剩余的可用带宽最大;$m \notin unavailableList$。
③ if 存在满足上述条件的节点 m
④ 将 m 添加至 *nodeList*,将 $l^{m,n}$ 添加至 *pathList*;
⑤ if $Cap(m)$ 大于等于 s_i^j 的资源需求
⑥ return *nodeList*, *pathList*;
⑦ else
⑧ 令 $n=m$;
⑨ end if
⑩ else
⑪ if *nodeList* = ∅ or *pathList* = ∅
⑫ s_i^j 无法部署
⑬ return;
⑭ else
⑮ 将 m 添加至 *unavailableList*,令 $n=tail(nodeList)$,删除 $tail(nodeList)$和 $tail(pathList)$
⑯ end if
⑰ end if
⑱ end for

当已部署了前一个 VNF 的服务器没有足够资源实例化当前 VNF 时,searchUp 算法被调用。该算法的作用是从服务器层向更高层逐层搜索满足条件的交换机,该交换机必须位于剩余带宽最多的链路上且 *Cap* 值大于等于 s_i^j 的资源需求。

3.3 算法设计

收益最大化的服务功能链优化编排算法步骤如下。

Step 1 动态资源配置。对所有的 SFC,求解优化问题(式(17)),得到资源最优配置方式,并将有解的 SFC 加入 *avaliableList*;

Step 2 VNF 放置。在 Step 1 的基础上,对所有 SFC,采用 searchUp 和 searchDown 算法确定待部署的服务器及映射的链路。

收益最大化的服务功能链编排算法伪代码如下。

算法 3 收益最大化的服务功能链编排算法。

输入:G、S、κ_i、ρ_i、D_i、r_i、min Φ_i^j、max Φ_i^j、min μ_i^j、min μ_i^j;

输出:SFC 配置及资源分配方案 *solution* (S)。

① 初始化输入参数, *serverId* 为当前部署的服务器编号,*switchId* 为交换机编号,*solution*(·)为部署解集;
② for all $s_i \in S$
③ 求解最优化问题(式(17))
④ if 存在最优解
⑤ add s_i into *avaliableList*
⑥ end if
⑦ end for
⑧ for all s_i in *avaliableList*
⑨ for all s_i^j in s_i
⑩ if $j=1$
⑪ 从数据中心根节点处调用 searchDown 算法返回服务器编号 *serverId*
⑫ else
⑬ if $n_{serverId}$ 满足 s_i^j 部署条件
⑭ 将 s_i^j 部署于 $n_{serverId}$, 更新 $solution(s_i^j)$
⑮ else
⑯ 调用 searchUp 返回交换机节点
⑰ if searchUp 有解

⑱ 获得 *switchId* 及对应的路由
⑲ else
⑳ $solution(S_i)=\varnothing$, 跳转至⑫
㉑ end if
㉒ 调用 searchDown 算法返回服务器节点
㉓ if searchDown 有解
㉔ 将 s_i^j 部署于 $n_{serverId}$,更新 $solution(s_i^j)$
㉕ else
㉖ $solution(S_i)=\varnothing$, 跳转至⑫
㉗ end if
㉘ end if
㉙ end if
㉚ end for
㉛ 返回部署方案 $solution(S_i)$
㉜ end for
㉝ 返回部署方案 $solution(S)$

3.4 算法复杂度分析

本文算法(算法 3)使用序列二次规划(sequential-quadratic-programming, SQP)[16-17]求解带约束的非线性优化问题(式(17))。假设 s_i^j 已部署在编号为 n'的主机内,需要求解 s_i^{j+1} 的部署位置。最坏的情况为遍历 n'和其他所有服务器之间的路径以找到符合部署条件的服务器及链路。当 pod 数为 k 时,服务器总量为 $k^3/4$, 任意两台服务器之间的最大路径数量为 k。考虑到网络拓扑层数为 4,因此在 s_i^j 部署位置已知的前提下,确定 s_i^{j+1} 的部署位置需要执行的搜索次数最多为 $4\times k\times k^3/4$。因此,算法的时间复杂度为 $O(N_r|S|k^4)$,其中,$|S|$为集合 S 中包含的 SFC 数量,N_r 为 SFC 包含 VNF 数量的平均值。

4 实验与结果分析

4.1 参数设置

通过 MATLAB 仿真实验对本文算法进行评估, 仿真参数设置如表 2 所示。实验对部署收益、部署成功率及资源利用率进行统计。部署成功率=成功部署的 SFC 数量/待部署的 SFC 总数;资源利用率=已使用的资源总量/(已使用服务器数量×服务器资源总量)。

SFC 编排过程中要求时延满足约束条件,本文将 SFC 分为会话、流式以及后台服务 3 类[5],每一类对于时延的要求分别为会话时延 ≤ 200 ms、流式时延 ≤ 300 ms、后台服务时延 ≤ 600 ms。实验中用到的 SFC 时延约束随机取其中一项。本实验分别取 pod 数为 4 和 6, SFC 数量为 10~100,其中 10~50 用于 pod 数为 4 的实验,60~100 用于 pod 数为 6 的实验。

实验所用处理器为 Intel(R) Core(TM) i7-8750H CPU@2.20 GHz,内存为 8 GB,操作系统为 Windows10 专业版。

表 2　实验参数设置

Table 2　Parameter setting in simulation

参数	取值	参数	取值
pod 数	4、6	r_i	[100,300]
res_n	1 000	VNF 种类数	10
$res_{m,n}^{bw}$	200	$\min \Phi_i^j$	[50,100]
$\|S\|$	10、20、30、40、50、60、70、80、90、100	$\max \Phi_i^j$	[150,250]
$\|s_i\|$	2、3、4	$\min \mu_i^j$	[200,400]
D_i	150、300、600	$\max \mu_i^j$	$\min \mu_i^j$+[100,200]
κ_i	[150,200]	σ	0.5
ρ_i	[10,40]	μ	0.5

4.2 实验结果

实验采用 Greedy 算法、ON_SFO 算法[15]、NF-LGT 算法[11]作为对比算法与本文算法进行比较以验证优化效果。ON_SFO 算法包含向上搜索与向下搜索 2 个阶段,但不包括回溯步骤。NF-LGT 算法第 1 阶段根据 worst-fit 策略选取距离当前节点时延最小的服务器部署 VNF,第 2 阶段通过交换服务器中的 VNF 实例寻找更优的部署方案。上述对比算法的资源分配取 $\max \Phi_i^j$。

图 2 为 pod 数为 4 时 4 种算法的对比效果。由图 2 可知,当 SFC 数量大于 30 时,由于计算资源和带宽资源的限制,对比算法的部署成功率下降较为明显,而本文算法在 SFC 数量增多时仍能保持较高的部署成功率。在部署收益方面,由于采用了弹性资源分配模型,本文算法在 SFC 数量较多时仍能保持较好的优化效果,与 Greedy 算法、NF-LGT 算法和 ON_SFO 算法相比,本文算法的最大提升分别为 67.2%、42.5%和 57.7%。在资源利用率方面,当 SFC 数量为 50 时,本文算法、Greedy 算法、NF-LGT 算法和 ON_SFO 算法资源利用率分别为 0.94、0.88、0.90 和 0.87,本文算法的资源利用率分别提高了 6.8%、4.4%、8.0%。这是由于采用了动态资源分配,能够以更加灵活的方式分配计算资源,提升资源利用率。

图 3 为 pod 数为 6 时 4 种算法的优化效果。当 SFC 数量为 60~100 时,本文算法、Greedy 算法、NF-LGT 算法和 ON_SFO 算法的部署成功率平均值分别为 0.886、0.711、0.743、0.719。在部署收益方面,与 NF-LGT 算法相比,本文算法在

SFC数量为60、70、80、90、100时的部署收益分别提高了28.8%、39.8%、49.9%、53.3%和48.2%。在资源利用率方面,当SFC数量为100时,本文算法、Greedy算法、NF-LGT算法和ON_SFO算法的资源利用率分别为0.93、0.88、0.90和0.88。

综上所述,本文算法在部署收益、部署成功率、资源利用率方面相较对比算法均有一定的提升,特别在SFC数量较多时具有一定优势。

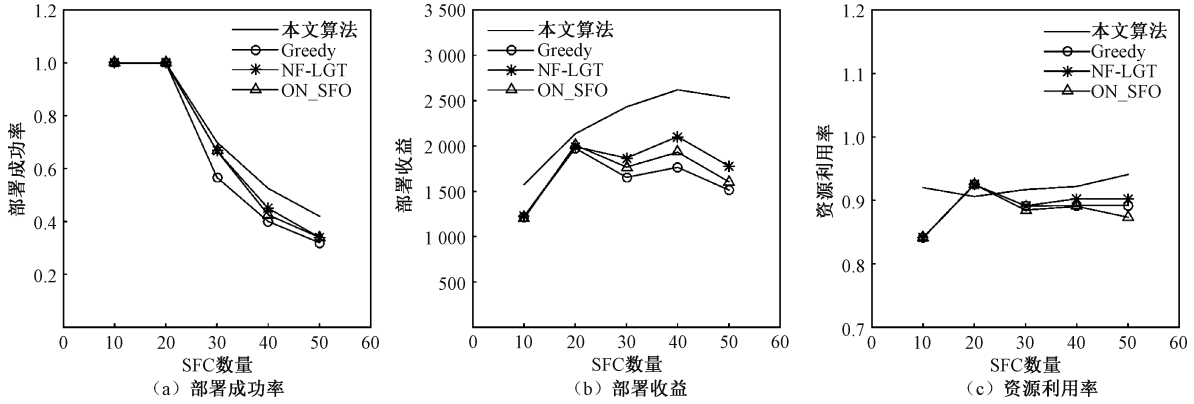

图2 pod数为4时的部署成功率、部署收益和资源利用率

Figure 2 Success rate, profit and resource utilization when pod is four

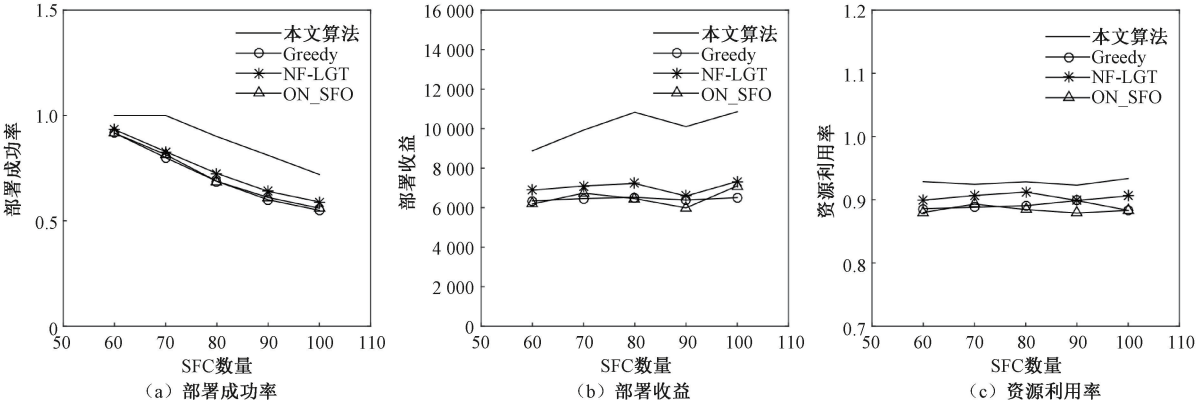

图3 pod数为6时的部署成功率、部署收益和资源利用率

Figure 3 Success rate, profit and resource utilization when pod is six

5 结论

本文以数据中心内的服务功能链编排作为场景,研究了请求到达率、计算资源与处理延时之间的关系,并以最大化部署收益为目标,提出基于弹性资源分配的服务功能链优化编排模型。在此基础上设计了一种两阶段启发式算法。仿真结果表明,本文提出的方法在优化服务链编排效果的同时能够兼顾求解效率及资源的合理利用,为改善基于NFV的网络服务能力提供了理论支持。

参考文献:

[1] GHAZNAVI M, SHAHRIAR N, KAMALI S, et al. Distributed service function chaining[J]. IEEE journal on selected areas in communications, 2017, 35(11): 2479-2489.

[2] HAN B, VIJAY G, JI L S, et al. Network function virtualization: challenges and opportunities for innovations [J]. IEEE communications magazine, 2015, 53(2): 90-97.

[3] MIJUMBI R, SERRAT J, GORRICHO J L, et al. Management and orchestration challenges in network functions virtualization [J]. IEEE communications magazine, 2016, 54(1): 98-105.

[4] AGARWAL S, MALANDRINO F, CHIASSERINI C F, et al. VNF placement and resource allocation for the support of vertical services in 5G networks[J]. IEEE/ACM transactions on networking, 2019, 27(1): 433-446.

[5] GIL-HERRERA J, BOTERO J F. Resource allocation in NFV: a comprehensive survey[J]. IEEE transactions on network and service management, 2016, 13(3): 518-532.

[6] BARI F, CHOWDHURY S R, AHMED R, et al. Orchestrating virtualized network functions[J]. IEEE transactions on network and service management, 2016, 13(4): 725-739.

[7] LI D F, HONG P L, XUE K P, et al. Virtual network function placement considering resource optimization and SFC requests in cloud datacenter[J]. IEEE transactions on parallel and distributed systems, 2018, 29(7): 1664-1677.

[8] LI Y, XUAN PHAN L T, LOO B T. Network functions virtualization with soft real-time guarantees[C]//The 35th Annual IEEE International Conference on Computer Communications. Piscataway:IEEE, 2016: 1-9.

[9] QIAO W X, LIU Y C, LU Y, et al. A novel approach for service function chain embedding in cloud datacenter networks[J]. IEEE communications letters, 2021, 25(4): 1134-1138.

[10] YU H F, CHEN Z R, SUN G, et al. Profit maximization of online service function chain orchestration in an inter-datacenter elastic optical network[J]. IEEE transactions on network and service management, 2021, 18(1): 973-985.

[11] THAI M T, LIN Y D, LAI Y C. A joint network and server load balancing algorithm for chaining virtualized network functions[C]//2016 IEEE International Conference on Communications. Piscataway:IEEE, 2016: 1-6.

[12] LIU Y C, LU H, LI X, et al. Dynamic service function chain orchestration for NFV/MEC-enabled IoT networks: a deep reinforcement learning approach[J]. IEEE internet of things journal, 2021, 8(9): 7450-7465.

[13] ALLEG A, AHMED T, MOSBAH M, et al. Delay-aware VNF placement and chaining based on a flexible resource allocation approach[C]//2017 13th International Conference on Network and Service Management (CNSM). Piscataway:IEEE, 2017: 1-7.

[14] ROCHER-GONZALEZ J, ESCUDERO-SAHUQUILLO J, GARCÍA P J, et al. Towards an efficient combination of adaptive routing and queuing schemes in fat-tree topologies[J]. Journal of parallel and distributed computing, 2021, 147: 46-63.

[15] 陈振荣. 数据中心网络中服务功能链的部署算法研究[D]. 成都: 电子科技大学, 2020.
CHEN Z R. Research on the algorithms for deploying service function chains in data center network[D]. Chengdu: University of Electronic Science and Technology of China, 2020.

[16] 陈宝林. 最优化理论与算法[M]. 2 版. 北京: 清华大学出版社, 2005.
CHEN B L. Optimal theory and algorithm[M]. 2nd ed. Beijing: Tsinghua University Press, 2005.

[17] 赵瑞安,吴方. 非线性最优化理论和方法[M]. 杭州: 浙江科学技术出版社, 1992.
ZHAO R A,WU F. Theory and methods for nonlinear optimization[M]. Hangzhou: Zhejiang Science & Technology Press, 1992.

A Profit Maximization Servies Function Chain Orchestration Algorithm

HUANG Hua[1,2],JIANG Jun[3],YANG Yongkang[2], HE Defeng[1],CAO Bin[4]

(1. College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China; 2. Eastern Communications Co., Ltd., Hangzhou 310053, China;3. College of Information Science and Electrical Engineering, Zhejiang Shuren University, Hangzhou 310053, China;4. College of Computer Science and Technology, College of Software, Zhejiang University of Technology, Hangzhou 310023, China)

Abstract: In this study, the service function chain (SFC) orchestration problem in data centers was investigated. The relationship between request arrival rate, computing resources and processing delay were analized. A flexible resource allocation optimization model aiming at maximize deployment benefits was proposed. Moreover, a heuristic method is developed to solve the problem in two stages. Firstly, the resource usage for SFC was optimized based on flexible resource allocation considering service delay. Secondly, for the virtual network function deployment and link mapping, the top-down and bottom-up search strategies based on worst-fit strategy were alternately adopted to improve the deployment efficiency and reduce the SFC delay. Finally, simulation with 4 and 6 pods in data center was designed to verify the performance of our heuristic algorithm. Experimental results demonstrated that, compared with existing methods, the development profit, success rate and resource utilization were improved in our algorithm.

Keywords: network function virtualization; service function chain; flexible allocation; combinatorial optimization