GAT-IL:一种基于图注意力网络与模仿学习的服务功能链部署方法

范琪琳[1],牛 岳[1],尹 浩[2],王天富[3],李秀华[1*],郝金隆[1]
(1. 重庆大学大数据与软件学院,重庆 400044;2. 清华大学北京信息科学与技术国家研究中心,北京 100084;
3. 中国科学技术大学计算机科学与技术学院,安徽合肥 230027)

摘 要: 网络功能虚拟化通过将网络功能从专用硬件设备迁移到商用服务器上运行的软件中间盒中,简化了网络服务的配置和管理. 在网络功能虚拟化的环境下,由一系列有序的虚拟网络功能组成的服务功能链正在成为承载网络服务的主流形式. 将底层物理网络资源分配给服务功能链的需求称为服务功能链部署问题. 对于基础设施提供商来说,在有限的资源条件下获得长期高回报是一个重要的挑战. 本文形式化定义了服务功能链部署问题,提出了一种基于图注意力网络与模仿学习的服务功能链部署方法(Graph Attention Network and Imitation Learning,GAT-IL). 该方法使用图注意力网络评估每个物理服务器的放置潜力,通过蒙特卡洛树搜索方法给出专家示范,并采用模仿学习方法进行智能体的训练,融入集束搜索策略优化解空间. 大量的实验结果表明,本文提出的GAT-IL方法在平均收益代价比和接受率的性能指标上均优于现有代表性算法.

关键词: 网络功能虚拟化;服务功能链;图注意力网络;模仿学习;蒙特卡洛树搜索;集束搜索
基金项目: 国家自然科学基金(No.62102053,No.61972222,No.92067206,No.62072060);国家重点研发计划(No.2019YFB1706101)
中图分类号: TP393.0 **文献标识码:** A **文章编号:** 0372-2112(2024)08-2811-13
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20221180

GAT-IL: A Service Function Chain Deployment Method Based on Graph Attention Network and Imitation Learning

FAN Qi-lin[1], NIU Yue[1], YIN Hao[2], WANG Tian-fu[3], LI Xiu-hua[1*], HAO Jin-long[1]
(1. *School of Big Data and Software Engineering*, *Chongqing University*, *Chongqing* 400044, *China*;
2. *Beijing National Research Center for Information Science and Technology*, *Tsinghua University*, *Beijing* 100084, *China*;
3. *School of Computer Science and Technology*, *University of Science and Technology of China*, *Hefei*, *Anhui* 230027, *China*)

Abstract: Network function virtualization simplifies the configuration and management of network services by migrating network functions from dedicated hardware devices to software middleboxes running on commercial servers. Under the environment of network function virtualization, the service function chain (SFC) composed of a series of ordered virtual network functions is becoming a mainstream alternative to host network services. The SFC deployment problem is to allocate the underlying physical network resources to the requirements of service function chains. It is challenging for infrastructure providers to obtain long-term high returns under limited resources. In this paper, we formally define the problem of SFC deployment and propose a novel method named graph attention network and imitation learning (GAT-IL) based on graph attention (GAT) network and imitation learning for SFC deployment. This method utilizes GAT to evaluate the potentials of each physical server, provides expert demonstrations through the Monte Carlo tree search algorithm, applies imitation learning to train the agent, and integrates the beam search strategy to optimize the solution space. Extensive experimental results show that the GAT-IL method proposed in this paper outperforms existing representative algorithms on performance metrics of average revenue-to-cost ratio and acceptance rate.

Key words: network function virtualization; service function chain; graph attention network; imitation learning; beam search

收稿日期:2022-10-20;修回日期:2024-01-21;责任编辑:王天慧
*通讯作者:李秀华

Foundation Item(s): National Natural Science Foundation of China (No.62102053, No.61972222, No.92067206, No.62072060); National Key Research and Development Program of China (No.2019YFB1706101)

1 引言

近年来,随着5G通信技术和物联网的快速发展,互联网的规模急剧增长.除此之外,多样化的应用需求,如基于流媒体的云游戏、基于图像和视频的虚拟现实、增强现实与远程医疗等也不断涌现.传统网络存在对资源调度和配置的灵活性不足、对新业务和应用的开通部署慢、对网络运营维护的要求高等一系列问题.在此背景下,网络功能虚拟化(Network Function Virtualization,NFV)新技术应运而生[1].NFV将网络功能从专用的硬件设备中解耦出来,允许通过软件化的虚拟网络功能(Virtualized Network Function,VNF)为用户提供多样化的网络服务[2].NFV技术让网络运营商可以方便地实现VNF的监控、管理和维护,降低网络功能的部署难度,提升网络性能和服务质量(Quality-of-Service,QoS).

近年来,基于NFV架构[3],服务功能链(Service Function Chain,SFC)成为学术界和工业界重点关注的对象.SFC由一系列按照特定顺序排列的VNF组成,它要求网络流按照顺序依次经过VNF的处理,从而实现特定的网络服务.服务功能链部署问题是指根据给定的优化目标将SFC中的VNF放置在合适的物理节点上,并选择相应的物理链路进行流量路由,同时需要满足资源或性能约束.由于该问题已被证明为NP难问题[4],许多学者致力于设计一些高效的方法来求解该问题.

基于启发式的算法通常利用一些人工规则来减少搜索空间,这在降低计算复杂度和运行时间方面通常是有效的.Liu等人[5]提出了一种利用节点排序方法辅助SFC部署的启发式算法,以获得最低的SFC部署成本.Luizelli等人[6]为了提高SFC部署的可扩展性,提出了一种基于二分搜索的启发式算法,目的是最小化部署到底层网络的VNF数量.Cheng等人[7]提出了研究服务链实例化的问题,并提出使用模拟退火算法来获得最优链路解.Yue等人[8]提出了一种两阶段启发式算法,目的是通过在一台服务器上尽可能多地部署SFC请求,并在多项式时间内找到部署方案,从而最小化网络总资源消耗.基于启发式的算法严重依赖于先验知识,通常考虑网络环境为非动态的情况.然而,现实中的网络环境是高度复杂和难以预测的.此外,基于启发式的算法通常是针对离线场景设计的,当运用到在线场景时,算法可能会停留在局部最优,远离全局最优,甚至可能会失效.

近年来,一些研究采用基于深度强化学习的算法处理组合优化问题.Tang等人[9]提出了一种基于Q学习的方法,并结合弹性伸缩技术来优化QoS.Xiao等人[10]提出了一种自适应的、基于策略梯度的深度强化学习算法,旨在联合优化SFC部署成本和SFC请求的总吞吐量.Wang等人[11]提出了一种基于双深度Q网络(Double Deep Q-Network,DDQN)的SFC部署方法,目的是最大化SFC接受率.Pan等人[12]通过利用时序差分(Temporal-Difference,TD)结合图卷积神经网络(Graph Convolution Network,GCN)模型提出了GCN-TD算法,目的是最大化互联网服务提供商(Internet Service Provider,ISP)的长期平均收益.然而,当强化学习获得不可行解时,模型可能会由强化学习反馈一个较差的奖励值,从而导致模型震荡,收敛变得缓慢.

上述工作都在努力解决服务功能链部署问题,但是针对在线服务场景,SFC请求随机到达,受到资源需求以及部署策略的影响,网络资源状态呈现出连续动态的复杂变化.如何在快速生成服务功能链部署方案的同时保证服务提供商的长期高收益代价比面临着挑战.因此,本文提出了一种基于图注意力网络(Graph Attention Network,GAT)与模仿学习(Imitation Learning,IL)的服务功能链部署方法GAT-IL.GAT-IL使用图注意力网络评估每个物理服务器的放置潜力,通过蒙特卡洛树搜索(Monte Carlo Tree Search,MCTS)方法给出专家示范,并采用模仿学习方法进行智能体的训练,融入集束搜索(Beam Search,BS)策略优化解空间.本文通过仿真实验,验证所提出算法的有效性.结果表明,本文提出的算法在平均收益代价比和接受率方面分别达到了0.953和96.60%,优于现有代表性算法.

2 系统模型与问题描述

2.1 物理网络模型

底层物理网络可以用无向加权图$G=(V,E)$来描述,其中,V表示物理服务器集合,E表示物理链路集合.本文选择物理服务器的计算能力和处理能力作为物理服务器的属性[13],选择物理链路的带宽和延迟作为物理链路的属性[14,15].H表示物理网络中所有VNF类型的集合.每个物理服务器$v\in V$,可以满足一组特定的VNF的嵌入,表示为$h_v(h_v\subset H)$,以及一定数量的存储资源容量S_v和计算资源容量C_v.不同的服务器可能提供类型一致的VNF(即$h_u\bigcap h_v$不为空).为简单起见,物理链路可以表示为$e_{u,v}\in E$,其中$u,v\in V$是其端

点. 本文使用$B_{u,v}$来表示物理链路$e_{u,v}$的总带宽. 此外,假设同一服务器v上的VNFs之间的带宽足以支持最大带宽,并允许在同一服务器上部署相同SFC请求下的不同VNF.

2.2 SFC请求模型

在特定的网络场景中,一个SFC请求需要流量有序经过一系列不同类型的VNF,每个VNF由底层物理网络中的服务器提供资源,从而实现特定的网络功能. 本文使用λ表示SFC请求的到达率,它服从泊松分布. 每个SFC请求R_t可以用有向加权图$R_t=(F_t,L_t)$表示,其中,$F_t=\{f_{t1},f_{t2},\cdots,f_{tM}\}$,表示为SFC请求$R_t$中的VNF集合. $L_t=\{l_{t1},l_{t2},\cdots,l_{tN}\}$,表示虚拟链路集合. 这两个集合中的$M=|F_t|$与$N=|L_t|$分别表示SFC请求$R_t$中VNF和虚拟链路的数量. 对于每一个VNF f_{tm}(第t个SFC请求对应的第m个VNF),用s_{tm}表示它的存储资源需求,用c_{tm}表示它的计算资源需求. b_t、i_t和o_t分别表示SFC请求R_t的带宽需求、源点和终点. a_t和p_t分别是SFC请求R_t的到达时间和生存期. SFC部署示例如图1所示,上部表示到来的SFC请求,包含了5个不同类型的VNF,VNF之间通过虚拟链路进行连接. VNF底部的数字表示处理能力需求和计算能力需求,虚拟链路周围的数字表示带宽需求. 图1下部表示物理链路层,在物理链路层中,每个物理服务器支持不同类型的VNF,VNF底部的数字表示该服务器上VNF的剩余资源.

2.3 问题描述

SFC部署问题可表示为映射:$R_t(F_t,L_t)\rightarrow G'(V',E')$,其中$V'\subset V,E'\subset E$. SFC部署过程由两个阶段组成:VNF节点映射和虚拟链路映射. 本文使用的关键符号及符号的含义如表1所示.

在VNF节点映射阶段,需满足以下约束条件:

$$x_{tm}^{v}=\begin{cases}1, & f_{tm}\text{已放置在服务器}v\text{上}\\ 0, & f_{tm}\text{未放置在服务器}v\text{上}\end{cases} \tag{1}$$

$$y_{tm}^{v}=\begin{cases}1, & h_v\cap f_{tm}\neq\varnothing\\ 0, & h_v\cap f_{tm}=\varnothing\end{cases} \tag{2}$$

$$\sum_{v}x_{tm}^{v}=1, f_{tm}\in F_t \tag{3}$$

$$x_{tm}^{v}\leqslant y_{tm}^{v},\forall f_{tm}\in F_t,\forall v\in V \tag{4}$$

$$\sum_{f_{tm}}s_{tm}\cdot x_{tm}^{v}\leqslant S_v,\forall f_{tm}\in F_t \tag{5}$$

$$\sum_{f_{tm}}c_{tm}\cdot x_{tm}^{v}\leqslant C_v,\forall f_{tm}\in F_t \tag{6}$$

式(1)表示物理服务器v上是否放置了VNF f_{tm}. 式(2)表示物理服务器v是否支持VNF f_{tm}类型的放置. 式(3)约束了每一个VNF不能放置在多个物理服务器v上. 式(4)是VNF类型约束,需要满足放置的VNF类型和物理服务器所容纳的VNF类型相同. 式(5)和式(6)分别是物理服务器的存储和计算资源约束.

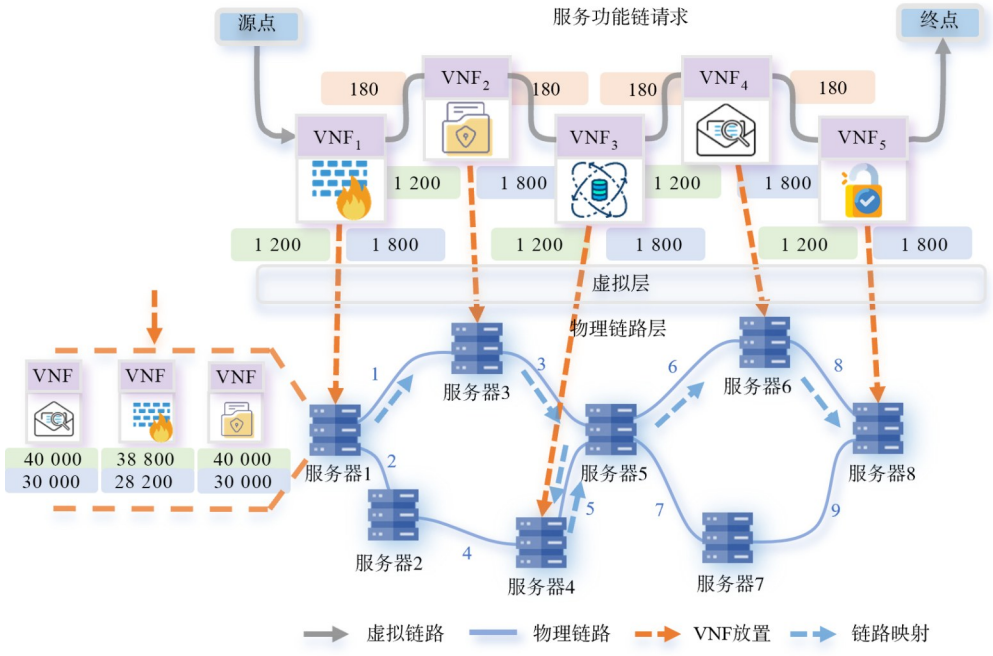


图1　服务功能链部署示例

表1 本文使用符号及含义

符号	含义	符号	含义
G	物理网络	p_t	R_t的生存期
V	物理服务器集合	x_{tm}^{v}	表示v上是否放置了f_{tm}
E	物理链路集合	$z_{tn}^{u,v}$	表示l_{tn}是否映射在$e_{u,v}$上
H	物理网络中的VNF类型集合	$I(v)$	v的入链路集合
v	物理服务器节点	$O(v)$	v的出链路集合
h_v	可嵌入v的VNF集合	$\mathrm{Y}(R_t)$	R_t的收益$\mathrm{R}(R_t)$与代价$\mathrm{C}(R_t)$的比值
C_v、S_v	v的计算、存储资源容量	α_S	存储资源的单位定价
$e_{u,v}$	uv间的物理链路	α_C	计算资源的单位定价
$B_{u,v}$	$e_{u,v}$的总带宽	α_B	带宽资源的单位定价
R_t	虚拟网络中的SFC请求	β_S	存储资源的单位成本
F_t	R_t中的VNF集合	β_C	计算资源的单位成本
L_t	R_t中的虚拟链路集合	β_B	带宽资源的单位成本
l_{tn}	第n条虚拟链路	$\boldsymbol{X}$,$\boldsymbol{X}'$	GAT中输入、输出的特征矩阵
M	虚拟网络中VNF数量	F,F'	GAT特征矩阵输入、输出的特征维数
N	虚拟网络中虚拟链路数量	$\boldsymbol{W}$	GAT中的权重矩阵
f_{tm}	第t个SFC请求对应的第m个VNF	K	GAT中注意力头数
s_{tm}、c_{tm}	f_{tm}的存储、计算资源需求	G_0	GAT的神经元个数
b_t	R_t的带宽需求	G_1	全连接层的神经元个数
i_t	R_t的源点	ξ	MCTS中UCT的探索次数
o_t	R_t的终点	η	BS中的集束带宽
a_t	R_t的到达时间	θ	训练好的GAT-IL参数

在虚拟链路映射阶段，本文给出了虚拟链路的映射约束，如下所示：

$$z_{tn}^{u,v}=\begin{cases}1, & \text{虚拟链路}l_{tn}\text{已映射在物理链路}e_{u,v}\\ 0, & \text{虚拟链路}l_{tn}\text{未映射在物理链路}e_{u,v}\end{cases} \tag{7}$$

$$\sum_{e_{u,v}\in E} z_{tn}^{u,v}\geqslant 0,\forall l_{tn}\in L_t \tag{8}$$

$$\sum_{l_{tn}\in L_t} z_{tn}^{u,v}\cdot b_t\leqslant B_{u,v},\forall e_{u,v}\in E \tag{9}$$

$$\sum_{e_{u,v}\in I(v)} z_{tn}^{u,v}-\sum_{e_{v,u}\in O(v)} z_{tn}^{v,u}=\begin{cases}-1, & v=i_t\\ 1, & v=o_t\\ 0, & \text{其他情况}\end{cases} \tag{10}$$

式(7)表示虚拟链路l_{tn}是否已映射在物理链路$e_{u,v}$. 式(8)表示一条虚拟链路l_{tn}，可能由不止一条的物理链路$e_{u,v}$构成. 式(9)表示的约束为链路带宽资源的取值约束，需要满足每一条物理链路$e_{u,v}\in E$上的带宽资源消耗不能超过总带宽容量. $I(v)$和$O(v)$分别表示物理节点v的入链路和出链路集合，式(10)的约束描述了对于每个SFC请求R_t的映射路径是一条从起点到终点不可分割的一条无环路径.

2.4 性能指标与目标函数

收益代价比可以很好地用于反馈SFC部署算法性能的优劣，具体定义如下：

$$\mathrm{Y}(R_t)=\frac{\mathrm{R}(R_t)}{\mathrm{C}(R_t)} \tag{11}$$

其中，收益$\mathrm{R}(R_t)$与三者有关：SFC请求所需的节点资源s_{tm}、c_{tm}，物理链路带宽资源b_t和SFC的生存周期p_t. 收益的定义表示如下：

$$\mathrm{R}(R_t)=\left[\alpha_S\cdot\sum_{f_{tm}\in F_t}s_{tm}+\alpha_C\cdot\sum_{f_{tm}\in F_t}c_{tm}+\alpha_B\cdot\sum_{l_{tn}\in L_t}b_t\right]\cdot p_t \tag{12}$$

其中，α_S、α_C和α_B分别是存储资源、计算资源和带宽资源的单位定价.

在SFC请求R_t完成部署后，部署过程所需的物理资源产生了部署代价$\mathrm{C}(R_t)$. $\mathrm{C}(R_t)$同样与三者有关：实际消耗的服务器存储资源s_{tm}、计算资源c_{tm}、链路带宽资源b_t和SFC请求R_t的生存周期p_t. 代价的定义表示如下：

$$\mathrm{C}(R_t)=\left[\beta_S\cdot\sum_{f_{tm}\in F_t}s_{tm}+\beta_C\cdot\sum_{f_{tm}\in F_t}c_{tm}+\beta_B\cdot\sum_{l_{tn}\in L_t}L(l_{tn})\cdot b_t\right]\cdot p_t \tag{13}$$

其中，β_S、β_C和β_B分别是存储资源、计算资源和带宽资源的单位成本，$L(l_{tn})$表示由虚拟链路l_{tn}映射的物理链路的长度.

本文用$\tilde{R}_T$表示T时刻以内接受的服务功能链请求

集合. 长期平均收益代价比如下所示:

$$\text{Avg\_RtC}=\lim_{T\to\infty}\frac{\sum_{R_t\in\tilde{R}_T}\text{Y}\left(R_t\right)}{\left|\tilde{R}_T\right|} \tag{14}$$

本文研究的部署问题的主要优化目标旨在最大化长期平均收益代价比,表示如下:

$$\max\ \text{avg\_RtC} \tag{15}$$
$$\text{s.t.}(1)\sim(10)$$

3　基于模仿学习的服务功能链部署算法

上述问题可以被归约为多维背包问题,该问题已被证明是NP难的[16]. 因此,本文拟设计一种基于图注意力网络与模仿学习的服务功能链部署方法执行在线决策.

3.1　基于模仿学习的服务功能链部署框架

模仿学习是通过示教者提供的范例学习,从而达到使智能体做出正确行为的目的. 行为克隆(Behavior Cloning,BC)是模仿学习的基础. 模仿学习作为一个类似监督学习的方式,首先由专家给出示范轨迹,模仿学习会通过BC在策略网络中学习专家的示范行为,然后基于已学习的策略再作出近似专家行为的决策. 由于在策略网络中学习的策略π_θ和专家示范策略π之间的误差会随时间增加而增大,随着误差的积累,可能导致后期模型训练低效,精度下降[17]. 针对这一现象,在策略网络和专家示范策略交互的环境中,本文使用数据聚合的方式[18],弥补这一局限性. 数据聚合从有限数据中产生额外的数据,实现数据样本量的增加来完成更多的训练,不断迭代该过程,直到模型收敛,实现误差最小化,以达到提升模型鲁棒性的目的.

图2给出了在SFC部署问题场景下的模仿学习模型示例. 该模型主要由专家(负责将状态S_t转换为示范动作 D_t)、智能体(根据专家给出的行为 D_t和环境传递的状态S_t做出动作A_t)和环境(使用数据聚合迭代,获取当前状态S_t,并通过A_t改变当前状态S_t为下一时刻状态S_{t+1})组成. 在本文所研究的问题中,环境是每次到来的SFC请求于底层物理网络状态.

3.2　基于图注意力网络结构的特征提取

本文使用图注意力(Graph Attention,GAT)网络模型来提取底层物理网络的特征信息. GAT是一种基于空间的图卷积网络,它的注意力机制是在聚合特征信息时,将注意力用于确定邻居节点的权重. 神经网络中注意力机制的加入可以帮助模型增加网络数据中重要信息的关注度,把更多的网络资源聚焦于这些重要信息,而并非同等关注整个网络[19]. $\mathcal{N}_v$代表v节点的邻居注意力节点集合. $\boldsymbol{X}=\left\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_{|V|}\right\}$是输入的特征矩阵,$\boldsymbol{x}_v\in\mathbb{R}^F$是物理节点$v$的输入特征,GAT模型输出的特征矩阵表示为$\boldsymbol{X}'=\left\{\boldsymbol{x}'_1,\boldsymbol{x}'_2,\cdots,\boldsymbol{x}'_{|V|}\right\}$,$\boldsymbol{x}'_v\in\mathbb{R}^{F'}$是物理节点$v$的输出特征,$F$和$F'$分别表示输入节点和输出节点特征的维数,$\boldsymbol{W}\in\mathbb{R}^{F'\times F}$是共享线性变换权重矩阵,将$\boldsymbol{W}\boldsymbol{x}_v$和$\boldsymbol{W}\boldsymbol{x}_u$进行$\|$拼接操作后得到的张量,使用前馈神经网络$\boldsymbol{a}\in\mathbb{R}^{2F'}$将其映射到实数上,便得到了未加工的注意力系数. 为避免梯度消失(vanishing gradients)和ReLU神经元"死亡"问题[20],本文选择LeakyReLU作为激活函数. 归一化后注意力系数a_{vu}的公式如下:

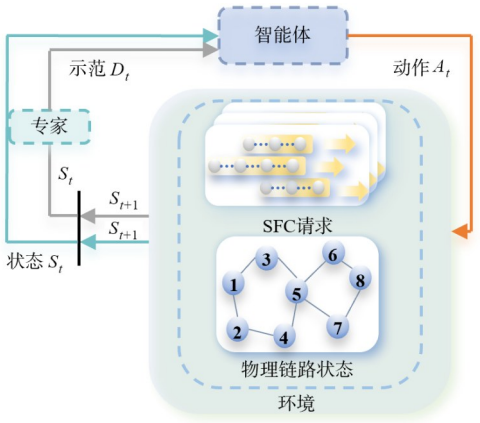


图2　基于模仿学习的服务功能链部署示例图

$$a_{vu}=\frac{\exp\left(\text{LeakyReLU}\left(\boldsymbol{a}^{\text{T}}\left[\boldsymbol{W}\boldsymbol{x}_v\|\boldsymbol{W}\boldsymbol{x}_u\right]\right)\right)}{\sum_{o\in\mathcal{N}_v}\exp\left(\text{LeakyReLU}\left(\boldsymbol{a}^{\text{T}}\left[\boldsymbol{W}\boldsymbol{x}_v\|\boldsymbol{W}\boldsymbol{x}_o\right]\right)\right)} \tag{16}$$

为了保证注意力机制的稳定性,本文使用了多头注意力(multi-head attention)机制. 多头注意力中节点v到节点u之间都有K个独立的注意力系数$a_{vu}=\left\{a^1_{vu},a^2_{vu},\cdots,a^k_{vu},\cdots,a^K_{vu}\right\}$,能够各自独立学习. 在实验过程中,注意力头数$K$的设置可根据预算时间大小和算法性能优劣进行权衡. 物理节点的输出特征计算公式如下:

$$\boldsymbol{x}'_v=\sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{u\in\mathcal{N}_v}a^k_{vu}\boldsymbol{W}^k\boldsymbol{x}_u\right) \tag{17}$$

加入了多头注意力的图注意力网络模型如图3所示.

3.3　基于蒙特卡洛树搜索的专家样本

本文使用蒙特卡洛树搜索(Monte Carlo Tree Search,MCTS)方法生成专家样本. MCTS根据采样生成搜索树,通过不断迭代着选择、扩展、仿真和回溯这4个过程,最终给出最佳策略,即选择出放置潜力最佳的物理服务器节点. 在MCTS算法中,分别用树的节点和与节点相邻的边表示在模仿学习阶段中产生的状态和动作. 随着算法迭代,更多节点的价值被估算出来,搜索树的规模逐步扩张,也使每次节点的估值变得更加

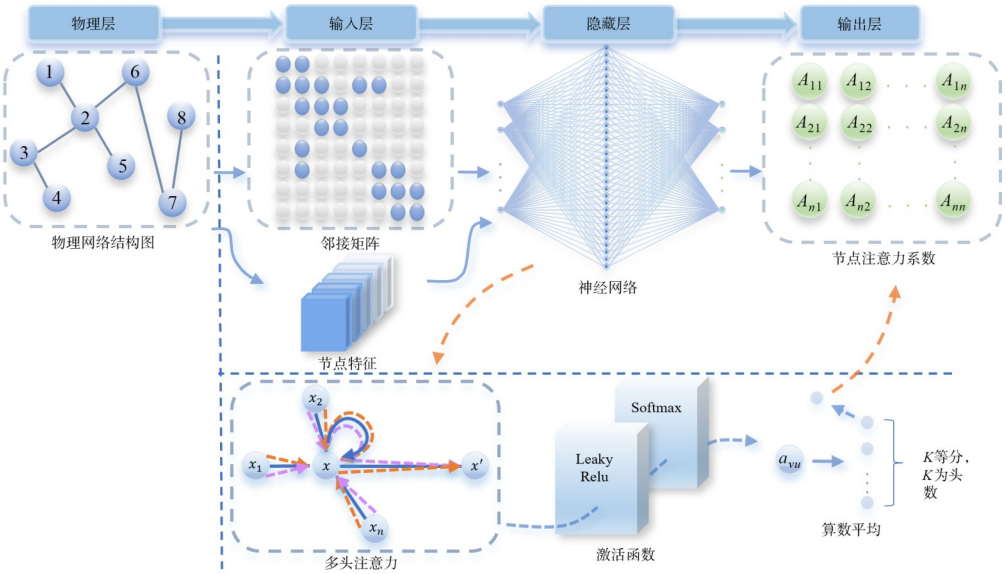


图3 图注意力网络模型

精确. MCTS 不断执行，直到算法条件超出给定的约束[21].

MCTS 的流程如图 4 所示，其主要可以分为 4 个步骤.

第 1 步：选择. 选择阶段的目的是找到未被搜索过的叶节点. 每次搜索根据置信树上限(Upper Confidence bounds for Trees，UCT)算法[22]规则进行搜索，节点选择公式如下：

$$u^* = \arg\max_{u \in U}\left(\frac{\gamma^u}{\rho^u} + \xi\sqrt{\frac{\ln\rho^v}{\rho^u}}\right) \tag{18}$$

其中，v 表示当前选择的节点；U 表示当前节点的子节点集合；ρ^v、ρ^u 分别表示当前节点 v 和子节点 u 被搜索的次数，γ^u 表示子节点的价值；ξ 表示探索系数，是一个常数；输出结果 u^* 表示最优 UCT 值对应的子节点.

第 2 步：扩展. 扩展阶段的目的是构建整个搜索树. 在扩展阶段中，如果被选择的节点不是叶节点(树的搜索尚未结束)，就生成一个子节点，用来存储子节点的数据信息. 扩展阶段结束后，MCTS 算法便进入第 3 步仿真阶段.

第 3 步：仿真. 仿真阶段的目的是进行随机动作的选择，模拟出各种结果. 根据由状态集随机产生的动作策略，从已选择的节点中进行仿真，直到到达叶节点(树的搜索结束)，停止仿真过程. 最终，仿真阶段会在某种输入对应的状态下根据动作策略，得到确定的结果[23].

第 4 步：回溯. 回溯阶段的目的是修正父节点估值. 回溯阶段将仿真估值得到的结果回溯至父节点，调整父节点的价值大小，使估值更加准确. MCTS 会沿着树的反向生成路径不断向上回溯，直至根节点. 回溯路径经过的每一个节点，其价值大小 γ^u 都等于在仿真阶段计算得到的奖励值. 此外，回溯路径经过的每一个节点的搜索次数 ρ^u，也会在这个过程中更新. 搜索次数代表节点的重要性. 一般来说，搜索次数越高，代表当前节点的价值越高，越应该被选择.

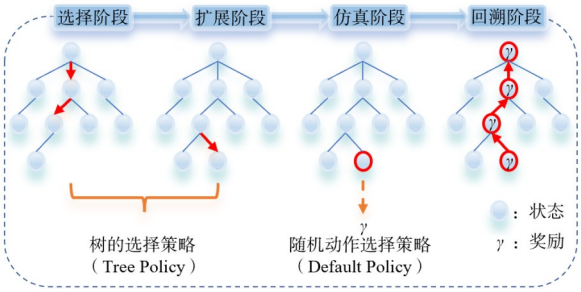


图4 蒙特卡洛树搜索示例

3.4 基于集束搜索算法的在线优化

对于较大搜索空间下的图结构，若采用传统的贪心搜索策略，SFC 部署策略可能会陷入局部最优的局面. 为此，本文采用集束搜索算法进行解空间优化. 集束搜索算法是一种启发式的图搜索算法. 算法对树的每一层利用广度优先搜索建立树节点，按照概率对节点进行从大到小排序，仅留下概率最高的 η 个节点，此 η 个节点即为集束带宽. 当 $\eta=1$ 时，表示贪心搜索算法. 每次保留下来的 η 个节点会在下一阶段继续扩展，其他节点则被剪枝. 最终可以得到 η 个序列，此 η 个序列对应了 η 个集束搜索路径，选择节点收益之和最高的路径，作为集束搜索的最终路径[24].

图 5 表示了在 SFC 部署问题的场景下 $\eta=2$ 的情况. SN_k 和 SN_{k+1} 分别表示当前时刻和下一时刻的候选服务器节点概率集. 利用集束搜索算法，分别对这两种

底层物理网络进行模拟. 迭代该过程,最终选出价值之和最高的路径,作为SFC部署决策.

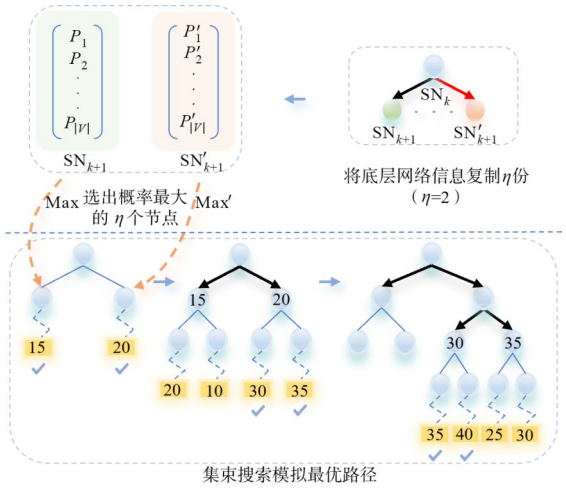


图5　集束搜索辅助服务功能链部署示例

3.5　GAT-IL: 基于图注意力网络与模仿学习的服务功能链部署方法

本文提出了名为GAT-IL的算法,它包括两个阶段:离线训练阶段和在线决策阶段. 与传统的SFC部署模型相比,本文提出的GAT-IL模型额外拥有MCTS专家示范、图注意力网络以及集束搜索的部分,这些模块的加入目的均是为了优化SFC部署算法的性能. GAT-IL算法的主要流程如图6所示. 整个算法主要分为4个步骤描述,包括阶段1(离线训练)中的步骤①、步骤②和步骤③与阶段2(在线决策)中的步骤④. 步骤①:将底层物理网络和到来的SFC请求信息进行特征提取. 步骤②:将提取的信息传入神经网络进行训练,计算节点注意力系数,经过神经网络的全连接操作并在归一化后得到节点放置概率,用参数θ表示. 步骤③:由MCTS提取底层网络特征和SFC状态后,给出专家示范,即生成标签. 训练智能体通过不断模仿专家行为,做出更加准确的决策. 同时,图注意力网络生成的参数也会随着专家给出的正确示范及时更新. 步骤④:将训练好的参数θ传入在线决策的模型中,计算节点放置概率,再利用集束搜索策略,进行快速准确在线决策. 最终得到SFC在物理网络上的部署结果.

(1)离线训练阶段:离线训练阶段需要提取节点(物理服务器)特征,归一化后加入到特征矩阵$\boldsymbol{X}$中. 节点特征包括已完成部署的VNF数量,当前节点的CPU剩余资源,当前节点的相邻链路剩余带宽总和,当前节点的相邻链路数量,当前节点与上一个已部署节点的链路距离. 首先,物理服务器的特征矩阵$\boldsymbol{X}$经过基于注意力机制的GAT模型输出$\boldsymbol{X}'$,再将结果传入全连接层. 具体做法是:通过矩阵变换,将$\boldsymbol{X}'$中的每个$\boldsymbol{x}'_v$和全连接层所有神经元节点进行连接,将原矩阵平铺成全连接神经元的维度大小. 使用全连接层的目的是进行特征数据的整合. 全连接后再通过矩阵变换,将生成的矩阵压缩成单维向量. 该向量的长度等于服务器节点个数. 然后,将向量中的所有元素归一化. 此时,该向量转化为了一个概率分布. 向量中的每个数值反映了服务器节点被选择的概率. 最后,判断服务器节点是否满足各种资源约束,筛选掉不符合条件的节点,并选择保留下来概率值最大的服务器节点作为输出. 由于MCTS已被证明在不计时间的情况下能够收敛于最佳性能[25]. 因此,在训练过程中,本文采用了MCTS驱动的模仿学

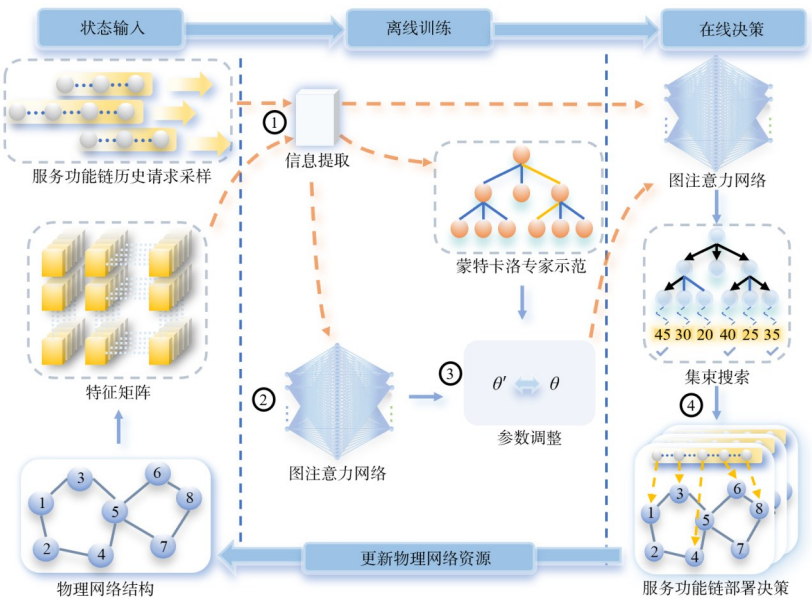


图6　GAT-IL算法流程

习算法. 神经网络参数通过MCTS给出的专家示范标签不断修正和优化. GAT-IL离线训练阶段算法如算法1所示.

算法1 GAT-IL离线训练算法

输入: 物理网络 $G=(V,E)$,SFC历史数据集dataSet

输出: 训练好的GAT-IL参数 θ

初始化GAT-IL模型
WHILE NOT GAT-IL模型收敛 **DO**
  从dataSet采样 H 个SFC连续请求构成训练集合 $R=\{R_1,R_2,\cdots,R_H\}$
  FOR $t=1$ **TO** H **DO**
    初始化树的根节点 root$\left(\gamma=0,\rho=0,\text{state}=\left(G,R_t\right)\right)$
    nodeMap $\leftarrow\varnothing$,vnI $\leftarrow 1$
    WHILE root非终止节点 **DO**
      $v\leftarrow$ root
      WHILE $\delta>0$ **DO**
        WHILE v非终止节点
          IF v完全扩展 **DO**
            $v\leftarrow$ 根据式(18)选择最优子节点
          ELSE
            $v\leftarrow$ 新建一个孩子节点,加入到搜索树中
            BREAK
        $u\leftarrow v$, $\Delta\leftarrow$ true
        WHILE u非终止节点 **DO**
          动作$a\leftarrow$随机选择一个物理服务器部署当前VNF
          IF 不满足资源约束 **DO**
            $\Delta\leftarrow$ false
            BREAK
          $u\leftarrow$ 状态转移$f(u,a)$
        IF $\Delta==$ true **DO**
          reward $\leftarrow$ 根据式(11)计算
        ELSE
          reward $\leftarrow 0$
        WHILE v 存在 **DO**
          $v.\gamma\leftarrow v.\gamma+$ reward, $v.\rho\leftarrow v.\rho+1$, $v\leftarrow v$.parent
        $\delta\leftarrow(\delta-1)$
      snI $\leftarrow \arg\max_{u\in U}\left(\frac{\text{root.childNode}[u].\gamma}{\text{root.childNode}[u].\rho}\right)$
      nodeMap.add(vnI, snI)
      vnI $\leftarrow$ vnI $+1$
      root = childNode[snI]
    IF nodeMap.size $==|F_t|$ **DO**
      使用Dijkstra计算节点间最短链路路径
      GAT-IL生成部署方案,并依据MCTS标签计算梯度和更新网络权重参数 θ
      更新物理网络状态 G
RETURN θ

(2)在线决策阶段:将离线训练阶段训练好的参数 θ 传递给在线决策阶段. 当SFC请求到达时,将底层物理网络状态、训练好的参数 θ 和集束带宽 η 作为在线阶段的输入,再利用集束搜索算法,每次根据集束搜索选择的节点,选出放置概率最大的 η 个服务器,再使用Dijkstra最短路径算法进行链路映射. 最终,得到节点收益之和最大的确定性路径,将其作为SFC部署方案. GAT-IL在线决策阶段算法如算法2所示.

算法2 GAT-IL在线决策算法

输入: 物理网络 $G=(V,E)$, SFC请求 $R_t=\left(F_t,L_t\right)$, 训练好的参数 θ, 集束带宽 η

输出: SFC部署策略

vnI $\leftarrow 1$, solutionList.add$\left((\varnothing,\varnothing,0)\right)$
WHILE vnI $\leqslant|F_t|$ **DO**
  nextSolutionList $\leftarrow\varnothing$
  FOR solution $\in$ solutionList **DO**
    $\boldsymbol{X}\leftarrow\boldsymbol{G}$ 在部署solution后的物理网络特征
    pVec $\leftarrow\boldsymbol{X}$ 传入参数为 θ 的神经网络得到可选物理节点的概率向量
    currList $\leftarrow$ 记录**pVec**中选出前 η 大 p 值对应的物理节点snI和 p 值
    FOR curr $\in$ currList **DO**
      nextSolution $\leftarrow$ solution $\cup$ (vnI, curr.snI, curr.p)
      nextSolutionList.add(nextSolution)
  solutionList $\leftarrow$ nextSolutionList保留收益之和最大的 η 个结果
  vn(I) $\leftarrow$ vn($I+1$)
nodeMapping $\leftarrow$solutionList中收益之和最大的结果
linkMapping $\leftarrow$Dijkstra计算节点间最短链路路径
RETURN nodeMapping, linkMapping

3.6 复杂度分析

在GAT-IL的离线训练阶段,计算复杂度与训练的数据量和训练次数相关. 经过离线训练后,GAT-IL可以使用训练好的模型来实现SFC的在线部署决策. 因此,计算复杂度只需关注在线场景即可. $O\left(|F_t|\cdot|V|\cdot\eta\right)$ 和 $O\left(|L_t|\cdot|E|\cdot\eta\cdot\log|V|\right)$ 分别为在线场景中选择候选节点和链路的计算复杂度.

在GAT-IL部署的前馈过程中,其计算复杂度主要与GAT-IL模型的规模有关. 设 G_0 表示GAT层的神经元个数,G_1 表示全连接层的神经元个数. GAT-IL前馈过程的计算复杂度为 $O\left(G_0\cdot|V|\cdot F+G_1\cdot\left(|V|+|E|\right)\right)$, 其中 $O\left(G_0\cdot|V|\cdot F\right)$ 为GAT层的计算复杂度,$O(G_1\cdot\left(|V|+|E|\right))$ 为全连接层的计算复杂度.

4　实验结果与对比分析

4.1　实验环境

本文的代码环境的构建采用的语言为Python. 实验环境如下:CPU采用Intel(R) Core (TM) i7-11700 CPU@2.50 GHz;编程语言版本为Python 3.7;深度学习框架采用TensorFlow 2.0;集成开发工具采用PyCharm 2021.2.2.

4.2　实验设置

实验过程中参数设置如下. 各资源的单位定价和单位成本都为1. 和底层物理网络模型$G=(V,E)$由工具GT-ITM生成. 该模型由60个物理服务器节点和150条物理链路组成. 每条物理链路$e_{u,v}\in E$其初始链路带宽$B_{u,v}$设置为1 Gbps,存储能力S_v设置为40 000,计算能力C_v设置为30 000. 一组包含有30个不同类型的VNF初始集合H被随机生成,每个物理服务器所能支持的VNF类型集合h_v是从已随机生成的集合H中再次随机选择的4个不同类型的VNF. 每个VNF $f_{tm}\in F_t$的CPU需求资源c_{tm}和带宽资源需求b_t之间存在联系,对于每个f_{tm},本文假定$c_{tm}=10b_t$或$c_{tm}=10b_t\cdot\ln b_t$,前后等式成立的概率分别为90%和10%[26];s_{tm}的值在[1 000, 4 000]范围区间随机生成. 每个SFC请求中的VNF集合R_t通过从已随机生成的集合H中再次随机生成,每个SFC中包含2到8个不等的VNF[27]. 此外,SFC请求$R_t=(F_t,L_t)$请求到达率遵循泊松分布,平均每秒到达一个SFC请求. 每个SFC请求的带宽需求b_t的取值在[150,190]之间均匀分布. 每个SFC请求的生存周期p_t设置为100 s,当SFC生存周期结束后,占用资源将被释放. 在满足SFC给定约束的情况下,随机生成2 000个SFC作为训练数据集. 在训练时,使用Adam优化器更新模型参数.

实验过程中GAT网络模型主要参数设置如下. GAT网络模型学习率α设置为0.000 25;训练时丢弃率dropout设置为0.5. GAT的神经元个数G_0设置为64;全连接层的神经元个数G_1设置为960. GAT和全连接的层数均设置为一层. 在GAT网络模型的特征矩阵中,输入节点和输出节点特征的维数F和F'均设置为5,输入和输出的特征设置为物理服务器的资源、相邻物理链路的数量、相邻物理链路的剩余带宽总和、与前一个已放置物理服务器间的路径长度、物理网络已部署的VNF数量. 网络模型批处理大小M'设置为100. MCTS探索系数ξ设置为0.5. 集束带宽η设置为2. 注意力头数K设置为3.

本文将对比以下算法来验证GAT-IL算法的有效性.

(1)TS:该算法是由Wang等人[28]提出的一种基于元启发式的禁忌搜索方法. 该算法首先选定一个可行解,然后在特定方向试探性地搜索,选择让函数值变化最多的目标作为下一步的路径方向.

(2)PL:该算法是由Fan等人[29]提出的一种基于位置优先级的启发式算法,旨在提高虚拟网络请求的接受率并优化物理网络资源利用率. 该算法首先考虑虚拟节点的位置和资源需求,然后通过路径综合评估来解决节点和链路映射之间的关联性问题,以及在虚拟网络映射过程中邻近节点资源消耗不平衡的问题.

(3)PACT:该算法是由Fu等人[30]提出的一种策略网络辅助蒙特卡洛树搜索的方法,通过使用卷积神经网络结合强化学习进行离线训练. 训练好的神经网络利用MCTS方法进行在线决策.

(4)GCN-TD:该算法是由Pan等人[12]提出的一种基于强化学习的算法,通过使用GCN结合TD学习对模型进行训练,训练后的模型对SFC执行部署决策.

(5)PG-CNN算法:该算法是由Yao等人[31]提出的一种基于强化学习的算法,通过反向传播的策略梯度(Policy Gradient,PG),用历史网络请求数据来训练含有卷积神经网络(Convolutional Neural Network,CNN)结构的策略网络,然后根据策略网络观察底层网络的状态并输出节点的映射结果,完成SFC的部署决策.

4.3　算法性能比较

(1)误差变化:图7显示了GAT-IL模型在训练过程中误差的变化. 从该图可以看出,模型输出的动作与专家给出的示范动作之间误差迅速减小并最终趋于稳定. 当神经网络进入训练时,会随机初始化一个参数值作为初始误差. 随着训练轮次的迭代,GAT-IL会根据专家示范不断修正误差. 由于本文使用了数据聚合的方式,刚开始训练时误差会快速下降. 在模型达到大约20个训练轮次时,误差逐渐收敛. 在模型达到大约40个训练轮次时,误差趋于稳定. 考虑到SFC的接受数量不再随着训练次数增加而增长,意味着此时神经网络的训练效果达到最佳.

(2)平均收益代价比对比:图8显示了在不断递增的SFC请求到达数量下的平均收益代价比. 从该图可以看出,所有算法的平均收益代价比都会随着逐渐递增的SFC请求数量而不断下降,这是因为可提供SFC请求的物理服务器的计算能力和处理能力逐渐减少,物理服务器的剩余资源已逐渐不能满足SFC的资源需求,导致成功部署SFC的代价增大. 这意味着平均收益代价比会随着累计到达的SFC请求数的增加而降低,从图中各曲线的下降趋势可以体现. 由于不同算法在物理网络候选集中选择物理服务器的成本代价不同,各算法会体现出性能的差异. 其中,PACT和PL算法表

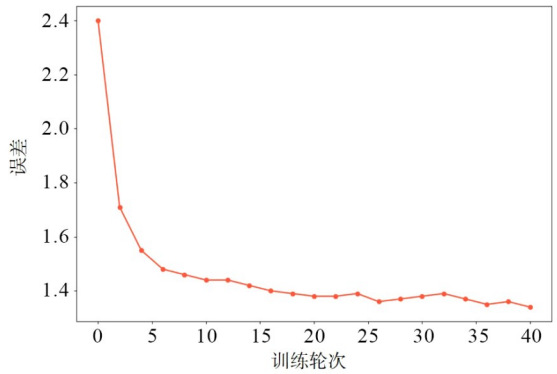


图7 GAT-IL训练过程误差变化

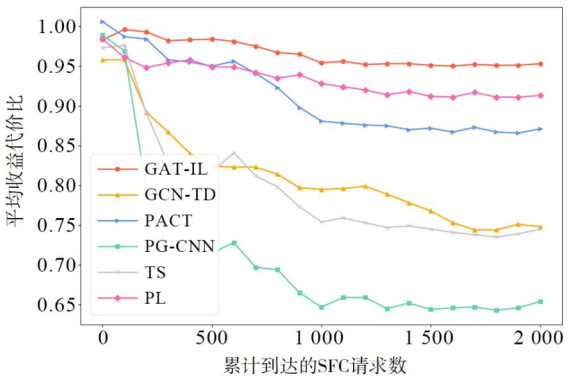


图8 平均收益代价比对比

现较好,这是因为PACT算法也参考了模仿学习方法,不断学习由专家示范给出的最优策略,使模型通过训练及时调整参数,从而表现出较高的收益.PL算法考虑了一种基于节点位置优先级的创新性方法完成映射过程,该算法可以较好地利用节点资源.然而,在不同的资源分配策略下,PL算法会更倾向于分配资源以提高接受率,或者更倾向于分配资源以提高总收益,从而导致此二者指标无法兼顾.本文提出的GAT-IL能够选出成本更低的候选节点,使物理服务器拥有更加充盈的资源容量,从而可以获得更高的收益代价比.当SFC请求累计到达数为2 000时,GAT-IL算法的平均收益代价比达到了0.953.GAT-IL算法分别比PL、PACT、GCN-TD、TS和PG-CNN算法高出4.38%、9.41%、27.41%、27.92%和45.72%.这是因为GAT-IL离线训练模型在MCTS给出的专家示范对应的情景下不断收敛,逼近了最优策略.此外,融合了集束搜索的GAT-IL模型在线决策方案,可以使VNF的放置结果更为合理,从而完成高效的SFC部署决策.

(3)接受率对比:在T时刻内,SFC请求的接受率可以定义为

$$\text{Acc\_Ratio}(T)=\frac{\left|\tilde{R}_T\right|}{\left|R_T\right|} \tag{19}$$

Acc_Ratio(T)与收益代价比类似,模型性能的提升同样对接受率的增长有较大的影响.图9显示了不同算法在SFC数量由0到2 000请求数量变化下的SFC部署接受率.当SFC请求累计到达数为2 000时,GAT-IL算法的SFC请求接受率达到了96.60%.GAT-IL算法分别比PL、PACT、GCN-TD、TS和PG-CNN算法高出31.79%、8.05%、20.37%、24.32%和40.31%.接受率指标和平均收益代价比指标类似,两指标均能反映算法对物理服务器剩余资源的利用效率.在刚开始有SFC请求到达时,各算法的接受率均为100%,这是因为此时物理服务器的资源充足,能完全满足VNF的资源需求.随着累计到达的SFC请求数的增多,物理服务器的剩余资源逐渐不能满足VNF的资源需求,接受率下降,各算法的差异也逐渐体现.不同算法在物理网络候选集中选择物理服务器的成本代价不同.这意味着算法若能以更低的成本接受SFC的部署,则消耗VNF的资源相对较少,物理网络的剩余资源会更加充足并且更多地SFC能够成功部署,反映在接受率的指标上,则体现出较高的接受率.实验结果表明,GAT-IL的性能高效,对底层物理网络资源的利用效率优于其他对比算法.

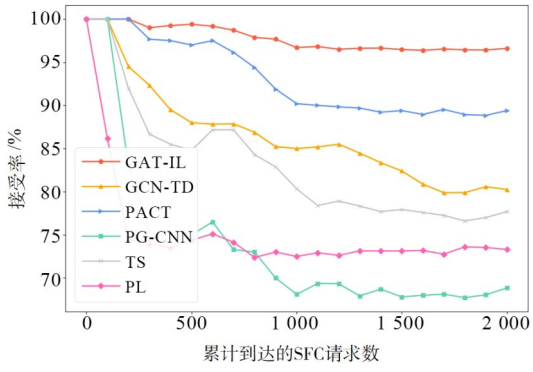


图9 接受率对比

(4)GAT模型的有效性:为了验证GAT模型的有效性,本文进行了消融实验,将GAT部分替换为GCN,并将对比算法命名为GCN-IL.图10显示了在SFC请求到达数量分别为500、1 000、1 500和2 000的情况下GAT-IL与GCN-IL的比较.两算法在指标为平均收益代价比的情况下作了比较.实验结果表明,GAT-IL算法性能优于GCN-IL算法,平均提高了3.67%.GAT-IL算法性能优于GCN-IL算法是因为GAT的注意力系数矩阵比GCN的拉普拉斯矩阵有更强的特征表达能力,节点特征之间的相关性被更好的融入到模型中.

(5)集束搜索的有效性:图11显示了在SFC请求到

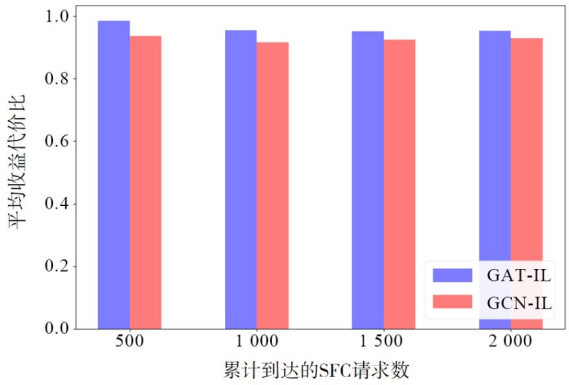


图 10　不同SFC请求总数下的神经网络模型性能对比

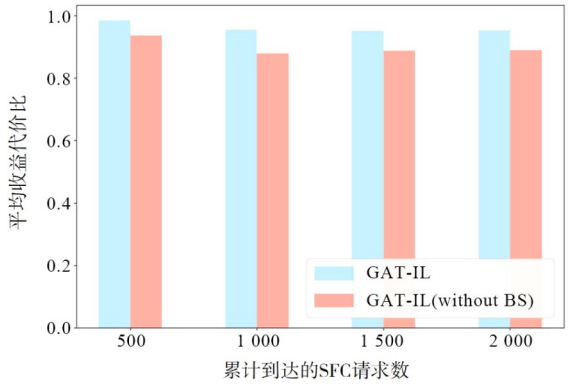


图 11　不同SFC请求总数下的算法性能对比

达数量分别为500、1 000、1 500和2 000的情况下，GAT-IL算法在有集束搜索和无集束搜索的平均收益代价比的对比．实验结果表明，GAT-IL算法性能优于GAT-IL(without BS)算法，平均提高了7.02%．集束搜索的加入，是在贪心算法的基础上，通过增大搜索范围，以获得比贪心算法更优的解．该实验结果验证了加入集束搜索的GAT-IL算法的有效性．

5　总结与未来工作

在SFC部署问题的情景下，本文提出了一种基于图注意力网络与模仿学习的服务功能链部署方法，称为GAI-IL算法，旨在最大化收益代价比．GAT-IL分为离线训练和在线决策两个阶段执行．在离线训练阶段，算法通过GAT提取物理网络和SFC请求的特征，利用模仿学习的方法，不断学习由MCTS给出的专家示范代表的最优策略，从而使模型通过训练及时调整参数．在在线决策阶段，算法根据离线训练阶段得到的深度神经网络，结合集束搜索算法进行决策，完成高效的SFC部署．最后，实验结果表明，本文提出的GAT-IL方法在平均收益代价比和接受率方面优于其他现有算法，证明了本文算法的有效性．在未来的工作中，我们将尝试用聚类的方法对服务功能链需求特性进行划分．此外，还考虑在训练过程中，将物理网络边信息进行特征挖掘和融入，从而设计基于物理网络资源与服务功能链需求特性双向感知的资源分配方法，实现更精准地SFC部署．

参考文献

[1] LI D F, HONG P L, XUE K P, et al. Availability aware VNF deployment in datacenter through shared redundancy and multi-tenancy[J]. IEEE Transactions on Network and Service Management, 2019, 16(4): 1651-1664.

[2] QU K G, ZHUANG W H, YE Q, et al. Dynamic flow migration for embedded services in SDN/NFV-enabled 5G core networks[J]. IEEE Transactions on Communications, 2020, 68(4): 2394-2408.

[3] MIRJALILY G, LUO Z. Optimal network function virtualization and service function chaining: A survey[J]. Chinese Journal of Electronics, 2018, 27(4): 704-717.

[4] COHEN R, LEWIN-EYTAN L, NAOR J S, et al. Near optimal placement of virtual network functions[C]//Proceedings of the International Conference on Computer Communications. Piscataway: IEEE, 2015: 1346-1354.

[5] LIU J Q, LI Y, ZHANG Y, et al. Improve service chaining performance with optimized middlebox placement[J]. IEEE Transactions on Services Computing, 2015, 10(4): 560-573.

[6] LUIZELLI M C, COSTA CORDEIRO W L DA, BURIOL L S, et al. A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining [J]. Computer Communications, 2017, 102: 67-77.

[7] CHENG G Z, CHEN H C, HU H C, et al. Enabling network function combination via service chain instantiation [J]. Computer Networks, 2015, 92: 396-407.

[8] YUE Y, CHENG B, LIU X, et al. Resource optimization and delay guarantee virtual network function placement for mapping SFC requests in cloud networks[J]. IEEE Transactions on Network and Service Management, 2021, 18(2): 1508-1523.

[9] TANG P, LI F, ZHOU W, et al. Efficient auto-scaling approach in the telco cloud using self-learning algorithm[C]// Proceedings of the Global Communications Conference. Piscataway: IEEE, 2015: 1-6.

[10] XIAO Y K, ZHANG Q X, LIU F M, et al. NFVdeep: Adaptive online service function chain deployment with deep reinforcement learning[C]//Proceedings of the International Symposium on Quality of Service. New York: ACM, 2019: 1-10.

[11] WANG L, MAO W X, ZHAO J, et al. DDQP: A double

deep Q-learning approach to online fault-tolerant SFC placement[J]. IEEE Transactions on Network and Service Management, 2021, 18(1): 118-132.

[12] PAN P, FAN Q L, WANG S, et al. GCN-TD: A learning-based approach for service function chain deployment on the fly[C]//Proceedings of the Global Communications Conference. Piscataway: IEEE, 2020: 1-6.

[13] LI B Y, CHENG B, LIU X, et al. Joint resource optimization and delay-aware virtual network function migration in data center networks[J]. IEEE Transactions on Network and Service Management, 2021, 18(3): 2960-2974.

[14] 陈卓, 冯钢, 刘蓓, 等. 运营商网络中面向时延优化的服务功能链迁移重配置策略[J]. 电子学报, 2018, 46(9): 2229-2237.
CHEN Z, FENG G, LIU B, et al. Delay optimization oriented service function chain migration and re-deployment in operator network[J]. Acta Electronica Sinica, 2018, 46 (9): 2229-2237. (in Chinese)

[15] JIN P, FEI X, ZHANG Q, et al. Latency-aware VNF chain deployment with efficient resource reuse at network edge[C]//Proceedings of the International Conference on Computer Communications. New York: ACM, 2020: 267-276.

[16] LAGHRISSI A, TALEB T. A survey on the placement of virtual resources and virtual network functions[J]. IEEE Communications Surveys & Tutorials, 2018, 21(2): 1409-1434.

[17] 林嘉豪, 章宗长, 姜冲, 等. 基于生成对抗网络的模仿学习综述[J]. 计算机学报, 2020, 43(2): 326-351.
LIN J H, ZHANG Z C, JIANG C, et al. A survey of imitation learning based on generative adversarial nets[J]. Chinese Journal of Computers, 2020, 43(2): 326-351. (in Chinese)

[18] KELLY M, SIDRANE C, DRIGGS-CAMPBELL K, et al. HG-DAgger: Interactive imitation learning with human experts[C]//Proceedings of the International Conference on Robotics and Automation. Piscataway: IEEE, 2019: 8077-8083.

[19] VELICKOVIC P, CUCURULL G, CASANOVA A, et al. Graph attention networks[J]. Stat, 2017, 1050: 20.

[20] MAAS A L, HANNUN A Y, NG A Y. Rectifier nonlinearities improve neural network acoustic models[J]. ICML Workshop on Deep Learning for Audio, Speech and Language Processing, 2013, 30(1): 3.

[21] HAERI S, TRAJKOVI Ć L. Virtual network embedding via Monte Carlo tree search[J]. IEEE Transactions on Cybernetics, 2017, 48(2): 510-521.

[22] VODOPIVEC T, SAMOTHRAKIS S, STER B. On Monte Carlo tree search and reinforcement learning[J]. Journal of Artificial Intelligence Research, 2017, 60: 881-936.

[23] MA X, DRIGGS-CAMPBELL K, ZHANG Z, et al. Monte-Carlo tree search for policy optimization[C]//Proceedings of the 28th International Joint Conference on Artificial Intelligence. California: International Joint Conferences on Artificial Intelligence Organization, 2019: 3116-3122.

[24] CAZENAVE T. Monte Carlo beam search[J]. IEEE Transactions on Computational Intelligence and AI in Games, 2012, 4(1): 68-72.

[25] SIRONI C F, LIU J, WINANDS M H M. Self-adaptive Monte Carlo tree search in general game playing[J]. IEEE Transactions on Games, 2018, 12(2): 132-144.

[26] KUO T W, LIOU B H, LIN K C J, et al. Deploying chains of virtual network functions: On the relation between link and server usage[J]. IEEE/ACM Transactions on Networking, 2018, 26(4): 1562-1576.

[27] 李茹杨, 彭慧民, 李仁刚, 等. 强化学习算法与应用综述[J]. 计算机系统应用, 2020, 29: 13-25.
LI R Y, PENG H M, LI R G, et al. Overview on algorithms and applications for reinforcement learning[J]. Computer Systems & Applications, 2020, 29: 13-25. (in Chinese)

[28] WANG W Z, HONG P L, LEE D F, et al. Virtual network forwarding graph embedding based on tabu search [C]//Proceedings of the 9th International Conference on Wireless Communications and Signal Processing. Piscataway: IEEE, 2017: 1-6.

[29] FAN W B, XIAO F, CHEN X B, et al. Efficient virtual network embedding of cloud-based data center networks into optical networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(11): 2793-2808.

[30] FU Z H, FAN Q L, ZHANG X, et al. Policy network assisted Monte Carlo tree search for intelligent service function chain deployment[C]//Proceedings of the 20th International Conference on Trust, Security and Privacy in Computing and Communications. Piscataway: IEEE, 2021: 1161-1168.

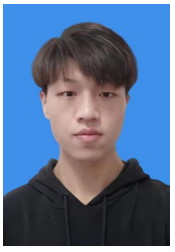
[31] YAO H, CHEN X, LI M, et al. A novel reinforcement learning algorithm for virtual network embedding[J]. Neurocomputing, 2018, 284: 1-9.

作者简介

范琪琳　女,1989年出生于江苏省无锡市.现为重庆大学大数据与软件学院副教授、硕士生导师.主要研究方向为边缘计算、深度学习.
E-mail: fanqilin@cqu.edu.cn

牛　岳　男,1995年10月出生于贵州省贵阳市.现为重庆大学大数据与软件学院硕士研究生.研究方向为虚拟网络嵌入.
E-mail: yuen@cqu.edu.cn

尹　浩　男,1974年10月出生于湖南省益阳市.现为清华大学北京信息科学与技术国家研究中心研究员、博士生导师.主要研究方向为计算机网络、大数据与区块链.中国电子学会会员编号:E1900006439S.
E-mail: h-yin@mail.tsinghua.edu.cn



王天富　男,2000年4月出生于山东省菏泽市.现为中国科学技术大学计算机科学与技术学院硕士研究生.主要研究方向为数据挖掘和网络资源管理.
E-mail: tianfuwang.cs@mail.ustc.edu.cn

李秀华　男,1987年4月出生于重庆市.现为重庆大学大数据与软件学院研究员.主要研究方向为边缘计算与缓存、边缘智能等.
E-mail: lixiuhua1988@gmail.com

郝金隆　男,1983年出生于河北省邯郸市.现任重庆长安科技有限责任公司智算与数据工程开发项目经理,应用软件开发高级架构师,重庆大学大数据与软件学院博士.主要研究方向为云端协同AI算力调度及应用.
E-mail: 20222401010G@stu.cqu.edu.cn