基于深度强化学习的软件定义网络 QoS 优化

兰巨龙，张学帅，胡宇翔，孙鹏浩
（国家数字交换系统工程技术研究中心，河南 郑州 450001）

摘　要：为解决软件定义网络场景中，当前主流的基于启发式算法的 QoS 优化方案常因参数与网络场景不匹配出现性能下降的问题，提出了基于深度强化学习的软件定义网络 QoS 优化算法。首先将网络资源和状态信息统一到网络模型中，然后通过长短期记忆网络提升算法的流量感知能力，最后基于深度强化学习生成满足 QoS 目标的动态流量调度策略。实验结果表明，相对于现有算法，所提算法不但保证了端到端传输时延和分组丢失率，而且提高了 22.7%的网络负载均衡程度，增加了 8.2%的网络吞吐率。

关键词：软件定义网络；深度强化学习；长短期记忆；服务质量
中图分类号：TP393
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2019227

Software-defined networking QoS optimization based on deep reinforcement learning

LAN Julong, ZHANG Xueshuai, HU Yuxiang, SUN Penghao
National Digital Switching System Engineering & Research Center, Zhengzhou 450001, China

Abstract: To solve the problem that the QoS optimization schemes which based on heuristic algorithm degraded often due to the mismatch between parameters and network characteristics in software-defined networking scenarios, a software-defined networking QoS optimization algorithm based on deep reinforcement learning was proposed. Firstly, the network resources and state information were integrated into the network model, and then the flow perception capability was improved by the long short-term memory, and finally the dynamic flow scheduling strategy, which satisfied the specific QoS objectives, were generated in combination with deep reinforcement learning. The experimental results show that, compared with the existing algorithms, the proposed algorithm not only ensures the end-to-end delay and packet loss rate, but also improves the network load balancing by 22.7% and increases the throughput by 8.2%.

Key words: software-defined networking, deep reinforcement learning, long short-term memory, quality of service

1　引言

近年来，互联网规模越来越大，网络应用越来越多，网络流量呈爆炸式增长。Cisco 公司的《思科视觉网络索引：预测和趋势》白皮书指出，2022 年全球网络总流量将达 4.8 ZB。面对如此庞大的网络需求，现有有限的网络资源正面临重大挑战。当前，以软件定义网络（SDN, software-defined networking）为代表的新型网络结构正促进互联网领域的重大变革。SDN 将原来高度耦合的控制逻辑与转发行为解耦，通过控制平面的开放接口获得全局网络视图，并提供给上层网络服务程序，实现了对全网的

收稿日期：2019–07–18；**修回日期**：2019–10–28
通信作者：张学帅，xshuai.zh@foxmail.com
基金项目：国家重点研发计划基金资助项目（No.2017YFB0803204）；国家自然科学基金资助项目（No.61521003, No.61702547, No.61872382）；广东省重点领域研发计划基金资助项目（No.2018B010113001）
Foundation Items: The National Key Research and Development Program of China (No.2017YFB0803204), The National Natural Science Foundation of China (No.61521003, No.61702547, No.61872382), The Research and Development Program in Key Areas of Guangdong Province (No.2018B010113001)

集中统一控制[1]。由于其在简化网络管理、降低运营成本、促进网络创新方面的卓越表现，SDN 架构现已被包括 Google、微软、Facebook 在内的世界著名互联网公司应用在内部网络建设中[2]。

因此，结合 SDN 的优势，充分发挥现有网络潜能，提高网络资源利用率，改善网络传输性能，对保证网络的服务质量有重要意义[3]。但 Hartman 等[4]指出在带宽受限的网络架构中寻找传输最大流量的链路集是 NP 完全问题，启发式算法仍是研究人员的首选思路。当前广泛采用的开路最短路径优先（OSPF, open shortest path first）算法和负载均衡（VLB, valiant load balancing）算法[5]，将所有流请求单一地路由到最短路径上或平均地路由到若干可用路径上，忽视了瓶颈链路的传输能力，难以取得最优结果。此外，在人工智能领域，深度强化学习（DRL, deep reinforcement learning）算法[6]借助深层神经网络[7]取得了突破性进展，而且已经有部分研究人员将 DRL 应用到网络控制场景中[8-9]。基于此，本文将长短期记忆（LSTM, long short-term memory）网络与 DRL 相结合，设计了软件定义网络的 QoS 优化机制，从而使 SDN 能够在满足 QoS 目标的基础上完成数据流调度。

具体而言，本文主要贡献包括以下 3 个方面。

1) 架构层面，提出了针对网络数据流的自动控制架构，实现了对数据流的自动化控制与调度。

2) 建模层面，通过建立软件定义网络 QoS 优化模型，引入经改进的深度强化学习算法对问题进行求解。

3) 实现层面，搭建了软件定义网络 QoS 优化实验环境，通过测试验证了算法的有效性。

2　相关工作

在 QoS 优化问题研究方面，现有主流方法可以分为启发式算法和机器学习算法 2 种，现对这 2 种方法总结概括如下。

2.1　启发式算法

Chen 等[10]为提高 SDN 云数据中心内部混合流的调度效率，设计了 Karuna 算法，在保证时延敏感性高的流传输的同时提高其他流的吞吐率；针对含有交叉节点和链路的网络流量 QoS 保障问题，Ongaro 等[11]设计提出了 MCFCSP（multi-commodity flow and constrained shortest path）算法，该算法先为多个流选择合适的路径，而后在满足链路带宽约束下将各流分到不同的路径中；Alizadeh 等[12]将拥塞控制和负载均衡联合考虑，针对网络流量的分层控制响应机制慢问题设计拥塞感知（CONGA, congestion aware）模型，将数据中心网络对拥塞问题的响应缩短到毫秒级。

以上算法主要通过建模来近似拟合当前的网络状态，并采用启发式方法为数据流请求实时计算路由配置，因此此类算法有严格的适用场景，链路带宽和长度的变化都可能导致状态拟合算法出现较大误差，进而对算法的整体性能产生显著影响。

2.2　机器学习算法

Xu 等[13]结合 SDN 架构和深度强化学习算法提出模型无关的流调度算法，利用网络自身的数据通过自学习的方式产生流控制逻辑，将一条数据流按照权重比例分配在选定的多条流通路上，避免建模过程中因近似处理带来的误差，具备良好的性能。但是本文分流（flow splitting）时采用的路径仍由 k 路最短路径（KSP, k-shortest path）算法产生，数据流仍会被分配在前 k 条较短路径上，难以避免链路瓶颈问题。在网络的拥塞控制方面，Xu 等[14]结合多路径传输控制协议（MPTCP, multi-path transmission control protocol）引入深度强化学习算法，以最大化网络利用率为目标，通过自学习实现了对 MPTCP 流的自动化拥塞控制。但是此算法只是在只有一台交换机的简单拓扑上进行了可行性验证，将算法扩展到较为复杂的网络拓扑中还有待进一步研究。

3　QoS 优化架构与建模

本节对基于深度强化学习的软件定义网络 QoS 优化算法中的具体元素进行阐述，并依据优化目标建立数学模型。

3.1　优化架构

本文提出的软件定义网络 QoS 优化架构的主要结构如图 1 所示，其中，各平面的具体介绍如下。

1) 数据平面。主要由一系列可编程交换机组成，接收控制器经南向接口发送控制策略并完成对数据分组的监测、处理和转发等操作，具体地，控制策略的载体可以是 OpenFlow 流表[15]或 P4 程序[16]。

2) 控制平面。负责连接应用平面与数据平面，维护全网视图，向下传递控制策略，向上将底层网络资源信息通过北向接口提供给应用平面。在北向接口选取上，基于 REST 架构的 RESTful[17]在响应时间方面表现出了特有优势，因此架构中采用

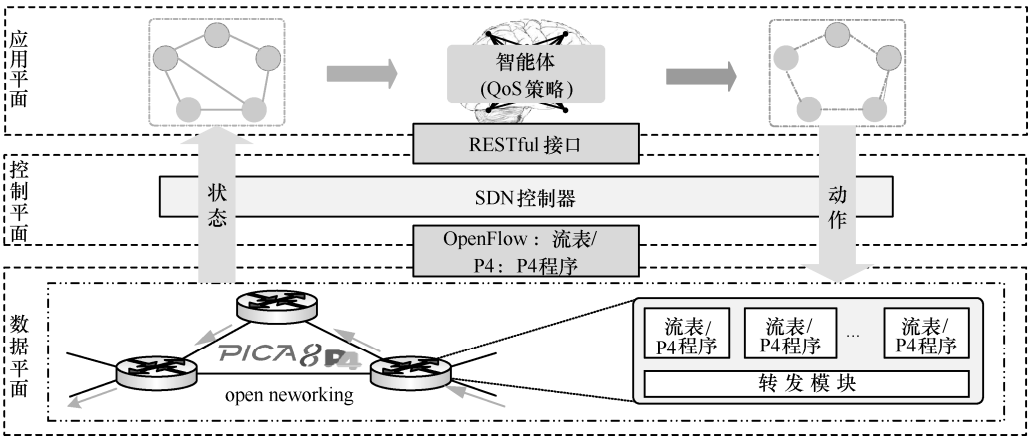

图 1 基于深度强化学习的软件定义网络 QoS 优化架构

RESTful 作为控制平面与应用平面的通信接口，以保证两平面间的通信效率。

3) 应用平面。运行负责网络 QoS 优化的强化学习智能体，智能体通过控制平面提供的全网逻辑视图和数据平面网络资源信息生成控制策略，而后将控制策略经北向接口向控制平面下发。

3.2 QoS 优化模型

1) 链路资源

SDN 拓扑模型记为无向图 $G=(V, E)$，V 为图 G 的节点集，代表实际网络中交换机集合；E 为图 G 的边集，代表实际网络中物理链路；网络链路用 e 表示，则 $e\in E$。系统中的每条网络链路的传输能力不尽相同，记 SDN 中链路 e 的带宽为 c_e。

2) 流量传输

节点集 V 中任意 2 个节点称作一组节点对，记 K 为所有节点对组成的集合。节点对间的一条路径记为一条转发路径 p，图 G 中所有节点对之间的所有转发路径组成的集合记为 P，节点对 k 间所有转发路径组成的集合记为 P_k。网络中所有节点对间的流请求矩阵记为 $\boldsymbol{D}$，对应的节点对 k 间的流请求记为 d_k。定义变量 w_k^p 表示节点对 k 间的流请求分配至路径 p 上的比重，变量 l_k 表示节点对 k 间的链路时延。

定义 1 链路利用率 u_e。链路利用率表示分配在链路 e 上的流量大小与链路带宽的比值，反映了链路的负载程度，如式(1)所示。

$$u_e=\sum_{p\in P}\sum_{e\in p}\sum_{k\in K}\frac{w_k^p d_k}{c_e} \tag{1}$$

定义 2 网络使用率 U。网络使用率定义为各链路利用率的最大值，如式(2)所示。

$$U=\max_{e\in E} u_e \tag{2}$$

定义 3 负载均衡程度 σ。用所有链路中链路利用率的最大值和最小值之差表示链路的负载均衡程度，可以表示为式(3)，因此 σ 越小，链路负载越均衡。

$$\sigma=\max_{e\in E} u_e-\min_{e\in E} u_e \tag{3}$$

基于以上分析，软件定义网络 QoS 优化问题可以建模为

$$\min \quad U \tag{4}$$

$$\text{s.t.} \quad \sum_{p\in P_k} w_k^p=1, \ \forall k\in K \tag{5}$$

$$u_e\leqslant U, \ \forall e\in E \tag{6}$$

$$w_k^p\geqslant 0, \ \forall p\in P, \forall k\in K \tag{7}$$

其中，式(4)为优化目标，即最小化网络利用率 U；式(5)表示网络节点对间的所有流请求都分配到节点对间的转发路径上；式(6)确保任意链路的链路利用率不超过网络使用率；式(7)表示分配至各转发路径上的流不能为负值。

4 算法设计

为解决上述 NP 完全问题，本节将引入基于深度强化学习的 QoS 优化算法，利用神经网络感知网络状态，并通过自学习的方法做出满足优化目标的分流决策。

4.1 强化学习算法

在一个典型的强化学习模型中，智能体与环境彼此交互状态（state）、动作（action）、奖赏（reward）

信息，通过训练逐步取得优化目标。具体来说，每一个步长 t，强化学习智能体获得当前环境的状态 $\boldsymbol{s}_t$，并根据策略函数 π 生成动作 $\boldsymbol{a}_t$，环境在执行动作 $\boldsymbol{a}_t$ 后，将自身状态 $\boldsymbol{s}_t$ 转变为 $\boldsymbol{s}_{t+1}$，同时将动作的奖赏 r_t 反馈给智能体。在这个过程中，四元组数据 $(\boldsymbol{s}_t,\boldsymbol{a}_t,r_t,\boldsymbol{s}_{t+1})$ 将作为经验保存在 Q 表[6]中，训练的目标就是利用 Q 表不断优化策略函数 π 的参数来最大化未来累积折扣奖赏 R_t 的期望 $\mathrm{E}(R_t)$。$\mathrm{E}(R_t)$可以表示为

$$\mathrm{E}(R_t)=\mathrm{E}\left(\sum_{t=0}^{T}\gamma^t r_t\right) \tag{8}$$

其中，T 为迭代次数，γ 是折扣因子，反映了当前动作对后续决策的影响程度。

在软件定义网络 QoS 优化问题中，状态、动作、奖赏的具体含义如下所示。

状态（state）。在此模型中，状态是指某一次网络测量[18]时网络中的流请求信息和所有链路的时延和利用率信息，用向量 $\boldsymbol{s}_t$ 表示，则 $\boldsymbol{s}_t=[\boldsymbol{D}_t,l_t,u_t]$，其中 $\boldsymbol{D}_t$ 表示 t 时刻的流请求矩阵，l_t 表示 t 时刻各节点对之间的传输时延，u_t 示 t 时刻网络中各链路的链路利用率。

动作（action）。动作是指强化学习智能体根据 QoS 策略函数和网络状态信息 $\boldsymbol{s}_t$ 生成的上述优化问题的解，也就是各节点对之间可用转发路径的分流比重。若 QoS 策略函数用变量 π 表示，动作向量 $\boldsymbol{a}_t$ 可由式(9)表示，其中 $\boldsymbol{a}_t=[w_1^1,w_1^2,\cdots,w_k^p,\cdots,w_{|K|}^{|P|}]$，$\boldsymbol{a}_t$ 中各分量的关系满足式(5)。

$$\boldsymbol{a}_t=\pi(\boldsymbol{s}_t) \tag{9}$$

奖赏（reward）。奖赏是对上一次动作所获收益的 QoS 评价。本模型中，软件定义网络 QoS 优化模型的优化目标是最小化网络使用率 U，为最大化奖赏符合优化目标，定义奖赏为 U 的相反数。当前 t 的奖赏 r_t 如式(10)所示，U_t 表示 t 时刻的网络利用率。

$$r_t=-U_t \tag{10}$$

4.2　QoS 优化算法

深度强化学习是深度学习和强化学习结合的产物，它综合了深层神经网络和强化学习的优势，利用神经网络代替传统 Q 表，具有更优的性能。将基于 Actor-Critic 架构的强化学习算法与神经网络相结合生成的 DDPG（deep deterministic policy gradient）算法[19]表现出优越的模型无关（model-free）特性，为强化学习在 SDN 的 QoS 优化问题中的应用提供了可能。在 DDPG 算法中共包含 4 个神经网络，即在线 Q 网络 $Q(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}^Q)$、在线策略网络 $\mu(\boldsymbol{s}|\boldsymbol{\theta}^\mu)$、目标 Q 网络 $Q'(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}^{Q'})$ 和目标策略网络。DDPG 算法把环境与智能体交互产生的四元组 $(\boldsymbol{s}_t,\boldsymbol{a}_t,r_t,\boldsymbol{s}_{t+1})$ 作为训练数据训练各 Q 网络和策略网络。为避免训练阶段的在线神经网络的参数 $\boldsymbol{\theta}^Q$ 和 $\boldsymbol{\theta}^\mu$ 变化幅度过大使 $\mu'(\boldsymbol{s}|\boldsymbol{\theta}^{\mu'})$ 无法收敛，DDPG 算法更新目标 Q 和目标策略网络的参数 $\boldsymbol{\theta}^{Q'}$ 和 $\boldsymbol{\theta}^{\mu'}$ 时采取平稳更新方法。

注意到，在网络场景中，网络状态常常随时间变化，具有明显的时间相关性，LSTM 在处理和预测表现出明显时间相关性的数据中取得了优良的成果[14]。本文在 DDPG 的基础上，引入 LSTM 层提出 R-DRL 算法，如图 2 所示。

在训练过程中，LSTM 网络负责对网络状态信息 $\boldsymbol{s}$ 进行预处理生成隐含状态 $\boldsymbol{h}$，并将该隐含状态传输给 Actor 和 Critic 架构中的神经网络，提高神经网络的决策的效率和准确性；Actor 和 Critic 架构中的神经网络依据 LSTM 网络提供的网络状态数据生成动作，并更新内部网络参数。具体来看，目标策略网络 $\mu'(\boldsymbol{s}|\boldsymbol{\theta}^{\mu'})$ 将参数 $\boldsymbol{\theta}^{\mu'}$ 传递给目标 Q 网络 $Q'(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}^{Q'})$ 用于函数评估，在线 Q 网络 $Q(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}^Q)$ 将参数 $\boldsymbol{\theta}^Q$ 传递给在线策略网络 $\mu(\boldsymbol{s}|\boldsymbol{\theta}^\mu)$ 用于该网络的梯度计算，R-DRL 算法的训练过程如算法 1 所示。

算法 1　R-DRL 算法训练过程

1) 初始化在线策略网络 μ、Q 网络 Q、LSTM 网络 R 和对应的目标网络

2) 初始化经验池 B

3) for each episode:

4)　初始化 Ornstein-Uhlenbeck 随机过程 O

5)　for $t=0$ to T:

6)　　根据 LSTM 网络 R 生成隐含状态 $\boldsymbol{h}_t$

7)　　生成动作：$\boldsymbol{a}_t=\mu(\boldsymbol{h}_t|\boldsymbol{\theta}^\mu)+\boldsymbol{O}_t$

8)　　执行动作 $\boldsymbol{a}_t$，观察 r_t，$\boldsymbol{s}_{t+1}$

9)　　将数据序列存入经验池 B

10)　　从 B 中随机采样取 M 个序列作训练用 $[\boldsymbol{s}_i,\boldsymbol{h}_i,\boldsymbol{a}_i,r_i,\boldsymbol{s}_{i+1}]$

11)　　通过目标网络 R' 获得 $\boldsymbol{h}_{i+1}$：$\boldsymbol{h}_{i+1}=R'(\boldsymbol{s}_{i+1})$

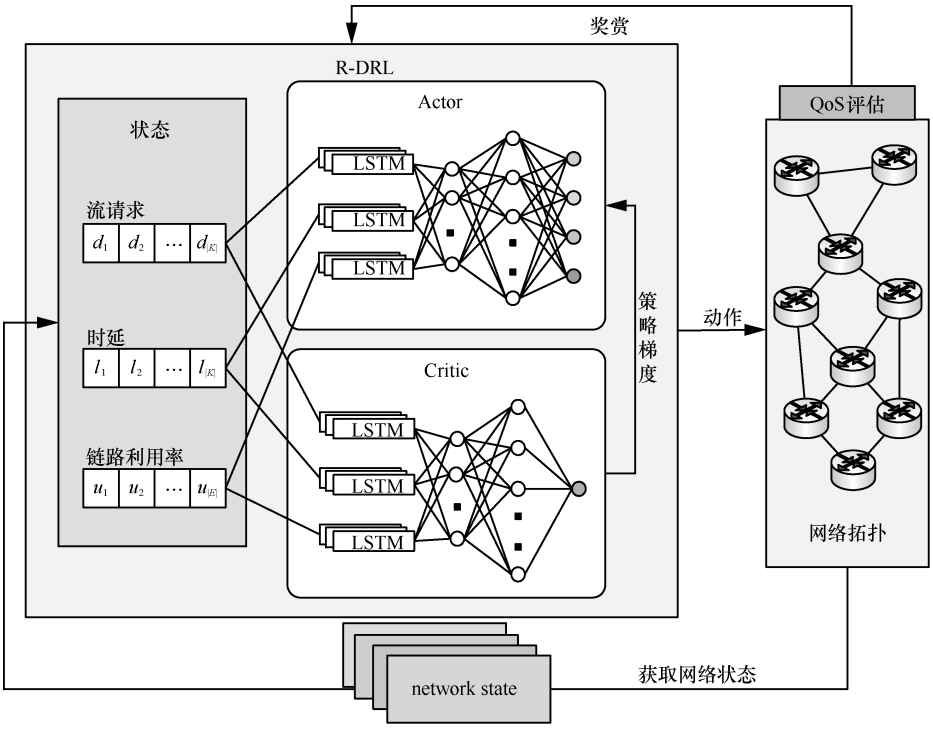

图 2 R-DRL 算法架构

12) 计算 Q 网络的输出值：$y_i = r_i + \gamma Q'(\boldsymbol{h}_{i+1}, \mu'(\boldsymbol{h}_{i+1} \mid \boldsymbol{\theta}^{\mu'}) \mid \boldsymbol{\theta}^{Q'})$

13) 更新在线 Q 网络，最小化误差函数：$L = \frac{1}{M}\sum_{i=1}^{M}(y_i - Q(\boldsymbol{h}_i, \boldsymbol{a}_i \mid \boldsymbol{\theta}^{Q}))^2$

14) 计算在线 Q 网络的策略梯度：$\nabla_{\boldsymbol{a}} Q(\boldsymbol{h}, \boldsymbol{a} \mid \boldsymbol{\theta}^{Q})|_{\boldsymbol{a}=\mu(\boldsymbol{h}\mid\boldsymbol{\theta}^{\mu}), \boldsymbol{h}=R(\boldsymbol{s}_i)}$

15) 更新在线策略网络：$\frac{1}{M}\sum_{i=1}^{M}\nabla_{\boldsymbol{\theta}^{\mu}}\mu(\boldsymbol{h} \mid \boldsymbol{\theta}^{\mu})\nabla_{\boldsymbol{a}} Q(\boldsymbol{h}, \boldsymbol{a} \mid \boldsymbol{\theta}^{Q})|_{\boldsymbol{a}=\mu(\boldsymbol{h}\mid\boldsymbol{\theta}^{\mu}), \boldsymbol{h}=R(\boldsymbol{s}_i)}$

16) 计算策略网络的梯度：$\nabla_{\boldsymbol{h}}\mu(\boldsymbol{h} \mid \boldsymbol{\theta}^{\mu})|_{\boldsymbol{h}=R(\boldsymbol{s}_i)}$

17) 更新在线 LSTM 网络：$\frac{1}{M}\sum_{i=1}^{M}\nabla_{\boldsymbol{h}} \cdot \mu(\boldsymbol{h} \mid \boldsymbol{\theta}^{\mu})\nabla_{\boldsymbol{a}} Q(\boldsymbol{h}, \boldsymbol{a} \mid \boldsymbol{\theta}^{Q})\nabla_{\boldsymbol{\theta}^{R}} R(\boldsymbol{s})|_{\boldsymbol{a}=\mu(\boldsymbol{h}\mid\boldsymbol{\theta}^{\mu}), \boldsymbol{h}=R(\boldsymbol{s}), \boldsymbol{s}=\boldsymbol{s}_i}$

18) 更新各目标网络

19) end for

20) end for

5 仿真评估

为验证 R-DRL 算法的性能，本节将通过实验仿真与其他方案进行对比，评价指标包括时延、分组丢失率和负载均衡程度 3 个方面。

5.1 实验方案

本文实验利用 Mininet 网络仿真平台来搭建虚拟 SDN，并采用开源 Ryu 控制器作为整个网络的控制器。采用 Iperf 流生成工具模拟真实网络流量，R-DRL 算法基于 Keras+TensorFlow 框架实现，算法和 Mininet 运行在一台服务器上，服务器具体配置如表 1 所示。

表 1 配置信息

属性	配置
操作系统	Ubuntu 16.04
CPU	Intel® Xeon E5
GPU	NVIDIA Tesla P100
内存	64 GB

实验中，SDN 的拓扑为认可度较高的 GEANT 网络，该网络中包含 22 个网络节点和 36 条链路，每条链路的传输能力各不相同。网络数据采用由 UPC 大学提供的基于 GEANT 拓扑收集的数据集，取 85%作训练集，15%作测试集。任意节点对之间的路由路径由生成树方法[20]生成，每个节点对之间采用 4 条备选转发路径。

5.2 对比方案

DDPG：DDPG 方案中不采用 LSTM 层，其余

神经网络参数设置与本文所提 R-DRL 算法一致，用来验证 LSTM 层对 DDPG 算法改进效果。

OSPF：OSPF 即开路最短路径优先，依据该规则，网络会把数据流转发在长度最短的路径上，由于没有考虑链路的传输能力，个别链路容易陷入拥塞。

MCFCSP：多物网络流流约束最短路径方案将链路的传输能力作为约束条件，在保证网络不出现拥塞的条件下传输数据流。

KSP：k 路最短路径方案会在两节点对间选择前 k 条最短的路径作为路由路径对数据流完成转发操作，本次实验中取 k=4。

5.3　实验结果

首先分析改变 LSTM 网络层数对 R-DRL 性能的影响，分别用 R_1、R_2、R_3 表示层数为 1、2、3 层的 LSTM 网络，R_0 表示不含 LSTM 层，即为 DDPG 算法。图 3 展示了训练过程中不同 LSTM 网络层数对 R-DRL 算法奖赏值 r_t 相对变化的影响。通过对比可知，使用 LSTM 可以明显提高算法的奖赏值，但不同的 LSTM 层数对 r_t 值的变化无明显影响。图 4 是不同 LSTM 层数的 R-DRL 算法的运行时间对比。当 LSTM 网络层数增加时，R-DRL 算法的所需的运行时间明显增加。结合图 3，为保证 R-DRL 算法的性能，以下实验取 LSTM 层数为 1。

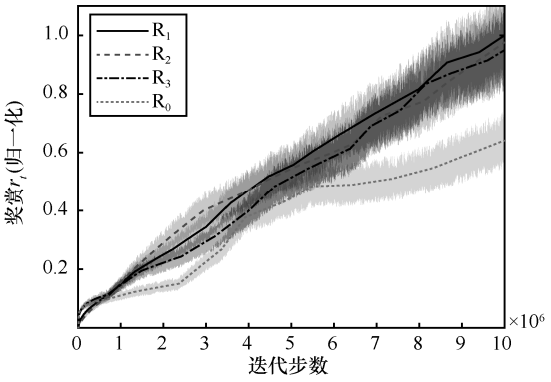

图 3　不同 LSTM 层数时奖赏 r_t 变化

图 5(a)展示了在软件定义网络 QoS 优化场景中不同数据流调度算法在时延方面的性能对比。从图 5(a)中可以看出，R-DRL 算法可以满足在流请求速率增长时网络的低时延需求，而且在时延方面的优化效果优于其他算法。DDPG 算法缺少 LSTM 网络层，对网络状态表现出的时间相关性无法有效感知，因此 DDPG 算法生成的优化策略不能准确反映网络的具体状态，陷入次优解；KSP 算法将数据流转发至较短的前 k 条路径上，因此这 k 条路径可能共用一条至多条较短的路径，由于传输能力限制，因此出现拥塞导致时延增加；MCFCSP、OSPF 算法和 KSP 情况一致，容易出现链路瓶颈问题。

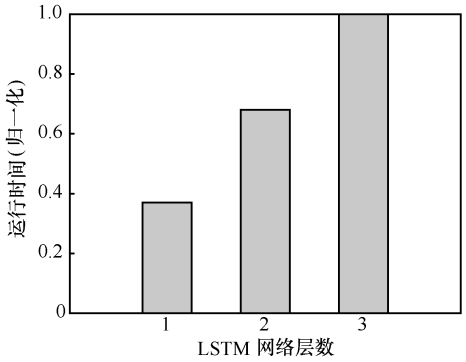

图 4　R-DRL 算法时运行时间对比

图 5(b)展示了在软件定义网络 QoS 优化场景中不同数据流调度算法在链路负载均衡方面的性能对比。从图 5(b)中可以看出，OSPF、MCFCSP 和 KSP 算法中衡量链路负载均衡程度的参数 σ 逐渐增大，相对于 R-DRL 和 DDPG 链路负载更不均衡，还可以注意到应用 R-DRL 算法的链路负载均衡程度总体要优于 DDPG 算法约 22.7%。

图 5(c)展示了不同数据流调度算法在分组丢失率方面的性能对比。从图 5(c)中可以看出，MCFCSP 以降低链路传输能力为代价保证各路径上的数据流不出现拥塞，分组丢失率最低，但 R-DRL 算法取得了接近于 MCFCSP 的性能。

图 6 联合采用箱线图和折线图展示了路由算法 R-DRL 和 KSP 对网络吞吐率的影响程度。其中折线图的连接点表示不同流请求速率条件下吞吐量数据的平均值，箱线图表示吞吐量数据的分布情况，箱线图的上下短横线表示箱线图上下限，叉号表示异常值。由于 DDPG、MCFCSP 和 OSPF 算法的吞吐率效果整体都没有超过 KSP，图 6 中省去了这 3 种算法的对比。通过图 6 可知，相对于 KSP 算法，R-DRL 将网络吞吐率提高了约 8.2%，这是因为生成树算法在生成路由路径时尽量减少各路径共用一条链路的情况出现，避免了瓶颈链路的产生，提高了网络的传输能力。

6　结束语

软件定义网络的 QoS 优化问题本质上是 NP 完

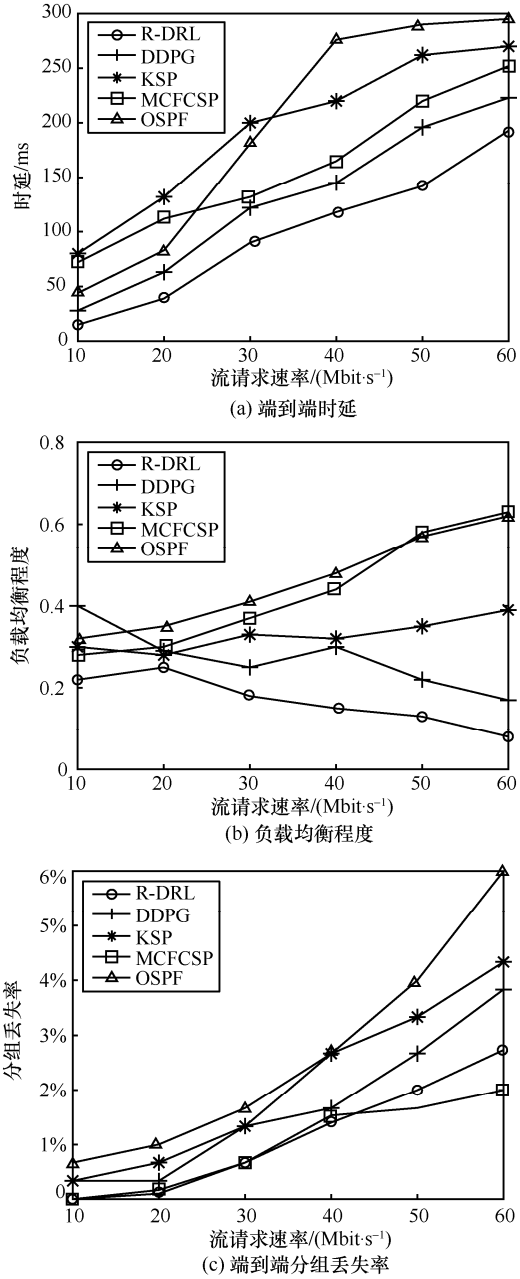

图 5 不同算法在软件定义网络 QoS 优化方面的性能对比

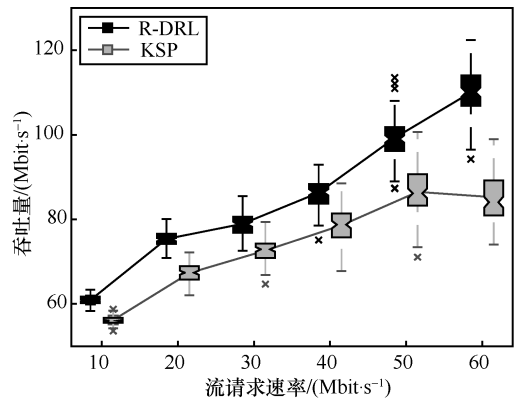

图 6 不同算法对 SDN 网络吞吐率的影响

全问题，为了实现 QoS 的自动化优化，国内外相关研究人员已经进行了深入研究，然而当前主流的启发式解决方案存在算法参数与场景绑定，应用场景受限问题，而在尝试应用深度强化学习的方案中，仍然采用 KSP 算法生成路由路径，难免出现链路瓶颈问题。基于此，本文借助长短期记忆网络和 DDPG 提出了用于软件定义网络 QoS 优化的 R-DRL 算法，通过将网络资源和状态信息统一到网络模型中，借助深层神经网络生成满足 QoS 优化的目标的动态流量调度策略。在网络拓扑 GEANT 上的仿真实验说明，结合长短期记忆网络设计的 R-DRL 算法提升了深度强化学习算法的流量感知能力，表现出比原有算法更好的性能。与其他算法相比，R-DRL 保证了传输时延和分组丢失率，提高了约 22.7%的网络的负载均衡程度和约 8.2%网络吞吐率，对解决 SDN 中的 QoS 优化问题有一定实用价值。

参考文献：

[1] MCKEOWN N. Software-defined networking[J]. INFOCOM Keynote Talk, 2009, 17(2): 30-32.

[2] 张朝昆, 崔勇, 唐墨祎, 等. 软件定义网络（SDN）研究进展[J]. 软件学报, 2015, 26(1): 62-81.
ZHANG C K, CUI Y, TANG H Y, et al. State-of-the-art survey on software-defined-networking (SDN)[J]. Journal of Software, 2015, 26(1): 62-81.

[3] 董谦, 李俊, 马宇翔, 等. 软件定义网络中基于分段路由的流量调度方法[J]. 通信学报, 2018, 39(11): 23-35.
DONG Q, LI J, MA Y X, et al. Traffic scheduling method based on segment routing in software-defined networking[J]. Journal on Communications, 2018, 39(11): 23-35.

[4] HARTMAN T, HASSIDIM A, KAPLAN H, et al. How to split a flow?[C]//2012 Proceedings IEEE INFOCOM. IEEE, 2012: 828-836.

[5] ZHANG S R. Valiant load-balancing: building networks that can support all traffic matrices[M]//London: Algorithms for Next Generation Networks. Springer, 2010: 19-30.

[6] SUTTON R S, BARTO A G. Reinforcement learning: an introduction[M]. MIT Press, 2018.

[7] LECUN Y, BENGIO Y, HINTON G. Deep learning[J]. Nature, 2015, 521(7553): 436.

[8] MAO H, NETRAVALI R, ALIZADEH M. Neural adaptive video streaming with pensieve[C]//The Conference of the ACM Special Interest Group on Data Communication. ACM, 2017: 197-210.

[9] XIAO S, HE D, GONG Z. Deep-q: traffic-driven QoS inference using deep generative network[C]//The 2018 Workshop on Network Meets AI & ML. ACM, 2018: 67-73.

[10] CHEN L, CHEN K, BAI W, et al. Scheduling mix-flows in commodity datacenters with karuna[C]//The 2016 ACM SIGCOMM Conference.

ACM, 2016: 174-187.

[11] ONGARO F, CERQUEIRA E, FOSCHINI L, et al. Enhancing the quality level support for real-time multimedia applications in software-defined networks[C]//2015 International Conference on Computing, Networking and Communications (ICNC). IEEE, 2015: 505-509.

[12] ALIZADEH M, EDSALL T, DHARMAPURIKAR S, et al. CONGA: distributed congestion-aware load balancing for datacenters[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(4): 503-514.

[13] XU Z, TANG J, MENG J, et al. Experience-driven networking: a deep reinforcement learning based approach[C]//IEEE INFOCOM 2018 IEEE Conference on Computer Communications. IEEE, 2018: 1871-1879.

[14] XU Z, TANG J, YIN C, et al. Experience-driven congestion control: when multi-path TCP meets deep reinforcement learning[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(6): 1325-1336.

[15] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, et al. OpenFlow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69-74.

[16] BOSSHART P, DALY D, GIBB G, et al. P4: programming protocol-independent packet processors[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 87-95.

[17] ZHOU W, LI L, LUO M, et al. REST API design patterns for SDN northbound API[C]//2014 28th International Conference on Advanced Information Networking and Applications Workshops. IEEE, 2014: 358-365.

[18] CLEMM A, CHANDRAMOULI M, KRISHNAMURTHY S. DNA: an SDN framework for distributed network analytics[C]//2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). IEEE, 2015: 9-17.

[19] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[C]//Eighth International Conference on Learning Representations. ICLR, 2016: 187-200.

[20] RÄCKE H. Optimal hierarchical decompositions for congestion minimization in networks[C]//The Fortieth Annual ACM Symposium on Theory of Computing. ACM, 2008: 255-264.

[作者简介]

兰巨龙（1962− ），男，河北张家口人，博士，国家数字交换系统工程技术研究中心教授、博士生导师，主要研究方向为未来信息通信网络关键理论与技术。

张学帅（1994− ），男，山东菏泽人，国家数字交换系统工程技术研究中心硕士生，主要研究方向为软件定义网络。

胡宇翔（1982− ），男，河南周口人，博士，国家数字交换系统工程技术研究中心副教授、博士生导师，主要研究方向为未来网络关键技术、网络智慧化等。

孙鹏浩（1992− ），男，山东青岛人，国家数字交换系统工程技术研究中心博士生，主要研究方向为软件定义网络、流量工程等。