QL-STCT：一种 SDN 链路故障智能路由收敛方法

李传煌，陈泱婷，唐晶晶，楼佳丽，谢仁华，方春涛，王伟明，陈超
（浙江工商大学信息与电子工程学院（萨塞克斯人工智能学院），浙江 杭州 310018）

摘　要：针对软件定义网络（SDN）链路故障发生时的路由收敛问题，提出了 Q-Learning 子拓扑收敛技术（QL-STCT）实现软件定义网络链路故障时的路由智能收敛。首先，选取网络中的部分节点作为枢纽节点，依据枢纽节点进行枢纽域的划分。然后，以枢纽域为单位构建区域特征，利用特征提出强化学习智能体探索策略来加快强化学习收敛。最后，通过强化学习构建子拓扑网络用于规划备用路径，并保证在周期窗口内备用路径的性能。实验仿真结果表明，所提方法能够有效提高链路故障网络的收敛速度与性能。

关键词：软件定义网络；链路故障；强化学习；路由收敛
中图分类号：TP393
文献标志码：A
DOI: 10.11959/j.issn.1000−436x.2022038

QL-STCT: an intelligent routing convergence method for SDN link failure

LI Chuanhuang, CHEN Yangting, TANG Jingjing, LOU Jiali, XIE Renhua,
FANG Chuntao, WANG Weiming, CHEN Chao
School of Information and Electronic Engineering (Sussex Artificial Intelligence Institute), Zhejiang Gongshang University, Hangzhou 310018, China

Abstract: Aiming at the problem of routing convergence when SDN link failure occurs, a Q-Learning sub-topological convergence technique (QL-STCT) was proposed to realize intelligent route convergence when SDN links fail. Firstly, some nodes were selected in the network as hub nodes and divides the hub domains according to the hub nodes, and the regional features were constructed with the hub domain as the unit. Secondly, the reinforcement learning agent exploration strategy was proposed by using the features to accelerate the convergence of reinforcement learning. Finally, a sub-topology network was constructed through reinforcement learning to plan the alternate path and ensure the performance of the alternate path in the periodic window. Experimental simulation results show that the proposed method effectively improves the convergence speed and performance of the link failure network.

Keywords: software defined network, link failure, reinforcement learning, routing convergence

0　引言

软件定义网络（SDN, software defined network）通过软件编程的形式定义和控制网络，极大地简化了网络的管理，促进了网络创新和发展。传统 IP 网络中的路由收敛是指同一网络拓扑下的路由器均需创建路由表，并通过与其他路由器交换拓扑信息以统一网络状态。而在 SDN 中，得益于数据平面与控制平面解耦和集中控制的优势，通过控制器监督、控制和管理网络并提供整个底层网络基础设施的实时视图[1]，实现高效的路由计算和对数据包的细粒度控制，快速响应链路和设备中发生的任何更改。

收稿日期：2021−11−17；**修回日期**：2022−01−17
基金项目：国家自然科学基金资助项目（No.61871468, No.61801427, No.62111540270）；浙江省新型网络标准与应用技术重点实验室基金资助项目（No.2013E10012）；浙江省重点研发计划基金资助项目（No.2020C01079）
Foundation Items: The National Natural Science Foundation of China (No.61871468, No.61801427, No.62111540270), The Key Laboratory of Network Standards and Applied Technology Foundation of Zhejiang Province (No.2013E10012), The Key Research and Development Program of Zhejiang Province (No.2020C01079)

尽管 SDN 相较于传统网络具有架构上的优势，但是链路故障问题依旧存在。当发生链路故障时，如何进行路由收敛，进行路由收敛操作时又将采用何种路由算法，目前并没有一个能充分发挥 SDN 的架构优势并受业界广泛接受的方案来专门应对 SDN 链路故障时的路由收敛需求，针对 SDN 的路由收敛问题一直是业界研究的热点[2-3]。

目前主流的 SDN 路由算法大多使用最短路径算法[4]。针对 SDN 中最短路径算法的缺点，文献[5]提出了一种多指标的链路负载均衡模型，该模型实时监视源节点到目的节点可用路径并采用基于多指标的综合评价算法评估检测到路径的性能，以此获得最优转发路径；文献[6]提出了一种基于 SDN 多路径并提供服务质量保证的系统（HiQoS, high-quality of service），其利用源节点与目的节点间存在多条路径的特性来保证不同类型流量的 QoS，由此保障链路故障发生时能快速恢复链路性能；文献[7]在 SDN 下采用改进最短路径算法并结合分离路径算法灵活控制流量。由于网络流量分布的不均衡增加了网络链路拥塞的可能性，文献[8]利用遗传算法搜索 SDN 全局网络视图中的优化路径提出了一种基于遗传算法的自适应 SDN 路由算法；文献[9]提出了一种基于多路径传输的动态负载均衡路由算法（DRAMP, dynamic routing algorithm based on multipath propagation）；文献[10]提出了一种基于 SDN 的负载均衡多路径路由算法。以上算法均以获得更好的网络性能为目标，充分利用网络中的冗余路径。

近年来，人工智能技术越来越受到研究者的关注，并运用强大的自我学习机制以实现最优。文献[11]提出了一种使用路由引擎在 SDN 的自治系统间（IAS, inter-autonomous systems）路由协议，并集成人工智能技术，实现灵活、高性能的路由能力，使网络获得良好的可伸缩性和收敛性。为更高效地适应动态网络环境，文献[12]提出了一种基于深度 Q 学习的路由策略来自动生成最优路由路径，并减少了 SDN 控制器策略的人工干预；文献[13]提出了一种新的路由路径优化算法串行粒子群优化（SPSO, serial particle swarm optimization）算法，并采用动态权重矩阵来提高路由性能；文献[14]提出了一种基于强化学习的路由规划算法，能有效地选择最优路径；文献[15]提出了一种强化学习路由算法来解决 SDN 在吞吐量和时延方面的流量工程问题，该算法使用网络吞吐量和时延作为奖励函数，经过适当的训练使智能体学习到最佳策略，通过预测网络行为获得最优路由路径。

实验表明，上述路由算法或方案在稳定的物理网络环境中规划全网路径时均具有较好的路由收敛效果，且在一定程度上提升了网络性能[8-15]。而在发生链路故障的网络环境中，采用重新规划全网路径的方式进行路由收敛将极大地消耗路径恢复时间，并影响正常链路中数据的实时转发[5,9,10,12,15]。因此需要更加快速且合适的路由收敛算法来应对链路故障的发生。本文提出了一种 SDN 中链路故障时的智能路由收敛方法 Q-Learning 子拓扑收敛技术（QL-STCT, Q-Learning sub topological convergence technique），该方法以实现枢纽节点竞选算法及基于枢纽节点的网络区域划分算法为基础，实现基于区域特征的子拓扑网络规划算法，并运用强化学习技术规划路径，通过仿真 SDN 链路故障情景，验证了所提出的强化学习模型对提高 SDN 链路故障下恢复路径性能的有效性。

1 系统模型

1.1 符号与定义

本节给出关于系统模型的相关定义。使用无向图 $G=(V,E)$ 表示网络拓扑结构，其中，V 表示所有网络节点的集合，E 表示网络拓扑中所有的链路集合。

定义 1 子拓扑网络。所选定 SDN 控制器管理的域内网络 G 中，由部分节点构建而成的网络，属于域内子拓扑网络，记为 L。该网络内嵌于 G，是由特定算法组合网络 G 中部分网络节点与链路形成的集合，即 $L\subsetneq G$，无论何时 L 均单个存在，并会根据网络环境变化进行变换，且不指代特定的网络节点与链路。

定义 2 枢纽节点。枢纽节点也称为枢纽交换机，是 SDN 域内各子拓扑网络 L 间实现数据交换的节点。$W=\{w_i \mid i=1,2,\cdots,n\}$ 表示枢纽节点的集合，i 表示编号，n 表示枢纽节点数量，枢纽节点 w_i 位于网络区域 i 的中心 $(i\in \mathrm{N}^+)$。

定义 3 边缘节点。网络 G 中除 w_i 以外的网络节点，也称为边缘交换机。$S=\{s_i \mid i=1,2,\cdots,n\}$ 表示边缘节点的集合，i 表示边缘节点编号，n 表示边缘节点数量，s_i 表示第 i 个边缘节点（$i\in \mathrm{N}^+$）。

定义 4 枢纽域。由单个枢纽节点 w_i 及部分通过算法筛选出的边缘节点组成的集合，记为 u_i。

$U=\{u_i \mid i=1,2,\cdots,n\}$ 表示枢纽域的集合，i 表示枢纽域编号，n 表示网络中枢纽域的数量，u_i 表示第 i 个枢纽域即枢纽域 $i(i\in \mathrm{N}^+)$。

定义 5　方向因子。用于描述相对于源节点位置路径转移方向的系数，记为 θ。在一条路径中，用 $\theta>0$ 描述远离源节点的过程，用 $\theta<0$ 描述靠近源节点的过程，用 $\theta=0$ 描述既不靠近也不远离源节点的过程。

1.2　系统模型介绍

系统模型如图 1 所示，在网络拓扑 G 中存在若干枢纽节点 w_i、边缘节点 s_i、节点间链路以及由枢纽节点与边缘节点包含链路组成的枢纽域 u_i。

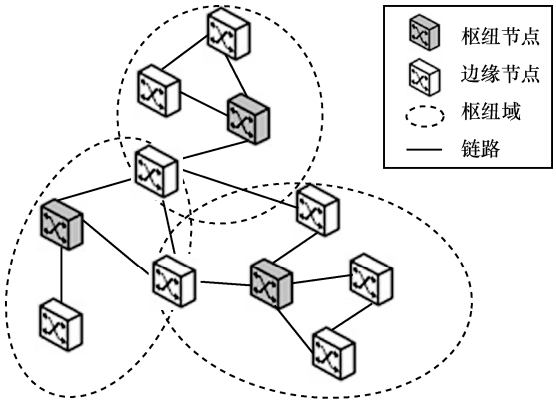

图 1　系统模型

网络节点集合 V 包含枢纽节点集合 W 以及边缘节点集合 S，即 $V=(W,S)$。网络整体包含多个枢纽域 u_i 形成枢纽域集合 U，一个枢纽域 u_i 包含若干个边缘节点 s_i 及单个枢纽节点 w_i，且枢纽域 u_i 间不能完全重叠，即 $u_i\neq u_j((i\neq j)\bigcap(i,j\in \mathrm{N}^+))$。从逻辑角度考虑，系统模型由若干个枢纽域连接形成，但实际上枢纽域集合永远不等于系统整体网络拓扑（$U\neq G$）。

在 SDN 控制器获得网络全局信息和故障信息后，SDN 智能路由收敛方案步骤描述如下。

步骤 1　竞选 w_i。通过控制器从数据平面获取信息 V 和 E，根据枢纽节点竞选算法及 E 和空间布局考虑，W 的产生必须满足 $W\subsetneq V$，即 $\exists s_i\in V$ 且 $s_i\notin W$。由此竞选出 W 的元素 w_i 用以构建子拓扑网络 L。

步骤 2　划分枢纽域。根据步骤 1 的结果，以 w_i 为一个网络枢纽域的核心，采用基于枢纽节点的网络区域划分算法为每个 w_i 选择对应的依附 s_i。当遍历完所有 w_i，完成网络枢纽域划分，获得枢纽域信息 U。

步骤 3　构建枢纽域特征，制定强化学习行为策略。根据步骤 2 的结果，以 u_i 为单位构建显性特征，引导强化学习智能体探索网络环境，制定高效的强化学习行为策略用于强化学习模型训练。

步骤 4　构建子拓扑网络 L。完成强化学习模型训练后，运用基于强化学习的子拓扑网络规划算法为 w_i 之间规划路径，实现 L 的构建，并通过周期性触发强化学习模型重新构建 L，保证在一个时间窗口内网络的收敛质量。当 L 的一条链路恰好为故障链路，同样触发 L 的重新构建，以保证收敛。

步骤 5　实现收敛。当 L 构建成功，路径信息以 Q 值表的形式保存。在接收到故障信息后，受故障影响的源节点和目的节点通过查询当前 L 的路径信息得到收敛路径。控制器向路径中对应的网络节点配置流表项，完成网络的收敛。

2　算法设计

2.1　枢纽节点竞选算法

枢纽节点竞选算法是对 SDN 控制器所管理的域内网络 G 中所有的网络节点进行链路连接数的计算，最终选取链路连接数较大的节点形成枢纽节点集合。网络重要节点相比于其他节点对网络的结构与功能有更大的影响[16]。枢纽节点作为网络区域的中心是其余节点进出子拓扑网络的出入口，在区域网络节点中具有最高影响力，其邻接节点越多，影响范围越大。因此，在网络拓扑中，具有较多邻接节点的网络节点可作为潜力枢纽节点。但由于拓扑中的链路均会发生故障，为使全网络节点能较快地连接并进出收敛网络，枢纽节点应该较均匀地分布于整个网络拓扑中，且过多的枢纽节点不仅会使子拓扑网络规模过大而丧失其原有功能，还会增加收敛时间和操作的流表数量，所以枢纽节点的数量应该加以控制。

因此，选择枢纽节点需要满足以下条件：1) 具有较多邻接网络节点；2) 在网络空间上较均匀分布；3) 控制数量择优选取。根据以上要求，设计算法 1，其中，$\boldsymbol{T}$ 为网络连接矩阵，node_links[n] 为每个节点的链路连接数记录参数，n 为每个节点的标识，hub 为枢纽节点集合，k 为对枢纽节点进行计数。

算法 1　枢纽节点竞选算法

1) for　i　in range (n)

2) node_link[i] $\leftarrow \sum_{j=0}^{j=n-1} \boldsymbol{T}[i][j]$

3) end for

4) while (1)

5) if max(node_link) == 2

6) break

7) end if

8) hub ←链路连接数最大节点

9) 将该节点链路连接数置 0

10) $k++$

11) for i in range (n)

12) if $\boldsymbol{T}$[hub(k)][i] == 1

13) node_links[i] $\leftarrow$ 0

14) end if

15) end for

在算法 1 中，通过网络拓扑的邻接矩阵计算出每个节点的链路连接数。为控制枢纽节点的数量，应选择具有较大影响性的节点作为枢纽节点，将链路连接数作为主要参考指标。链路连接数越大的节点，影响性越大，相反则影响性越小。因此，算法 1 需选出具有最高链路连接数的节点作为枢纽节点。

此外，为使枢纽节点均匀分布于网络中，在选出一个具有最高链路连接数的节点作为枢纽节点后，将枢纽节点及其所有的下一跳节点的链路连接数置 0，并在剩余节点中挑选下一个枢纽节点，重复上述操作。

由于算法 1 中将枢纽节点及其所有下一跳节点的链路连接数置 0，当算法运行到拓扑中只剩下链路连接数最大为 2 的孤岛节点时，该类孤岛节点的下一跳节点也是另一个枢纽节点的下一跳节点。为控制枢纽节点的数量，这类节点将不会作为枢纽节点。

通过算法 1，最终输出经过数量控制并均匀分布于网络中的枢纽节点集合W。

2.2 基于枢纽节点的网络区域划分算法

基于枢纽节点的网络区域划分算法是以筛选出的枢纽节点集合为基础，依据网络连接矩阵确定每个枢纽节点的依附节点，最终实现网络区域划分。在网络拓扑中，选择枢纽节点为中心划定枢纽域。当收敛事件发生时，通过事件位置确定其所在枢纽域，并重新规划路径绕开故障区域，完成源区域到目的区域的规划。依据枢纽节点位置对全网络进行合理的粗粒度区域划分，网络收敛时，先在区域尺度上完成路径的粗粒度规划，同时实现故障规避，使网络收敛行为更加高效快捷。

枢纽节点竞选算法中将枢纽节点及其所有下一跳节点的链路连接数置 0，即枢纽节点的所有下一跳节点成为该枢纽节点的依附节点，从而组成一个枢纽域，基于枢纽节点的网络区域划分算法如算法 2 所示，其中，$\boldsymbol{T}$ 表示网络连接矩阵，hub 表示枢纽节点集合，n 表示枢纽节点数量，nodes 表示网络节点集合，links 表示链路集合，not_hub 表示非枢纽节点的网络节点集合，area_nodes[n] 表示区域网络节点集合，area_links[n] 表示区域链路集合，$\boldsymbol{A}[n\times n]$ 表示区域连接矩阵，node_belongto_area[n] 表示节点依附区域集合。

算法 2 基于枢纽节点的网络区域划分算法

初始化 区域网络节点集合、区域链路集合、区域连接矩阵和节点依附区域集合

1) for i in hub

2) for j in nodes

3) if $\boldsymbol{T}[i][j] == 1$

4) area_nodes[i] $\leftarrow j$

5) area_links[i] $\leftarrow$ link[i][j]

6) node_belongto_area[j] $\leftarrow i$

7) 从 not_hub 中移除节点 j

8) 从 links 中移除链路 link[i][j]

9) end if

10) end for

11) end for

12) while not_hub != ϕ

13) for k in not_hub

14) if k 的连接节点都未选定区域

15) continue

16) else

17) 连接节点所在区域节点集合 $\leftarrow k$

18) 对应区域链路集合 $\leftarrow$ 相关链路

19) node_belongto_area[k] $\leftarrow$ 对应区域

20) 从 not_hub 中移除节点 k

21) 从 links 中移除对应链路

22) end if

23) end for

24) end while

25) for i in nodes

26) temp $\leftarrow$ node_belongto_area[i]

27) while (len(temp) > 1)

```
28)     for  j  in  temp[1:]
29)         A[hub.index(temp[0])]
            [hub.index(temp[j])] ← 1
30)         temp.remove(temp[0])
31)     end for
32)   end while
33) end for
```

算法 2 选取枢纽节点的依附节点，实现网络拓扑的粗粒度区域划分，并输出各枢纽域间的连接矩阵 $\boldsymbol{A}$ 以及交界节点和交界链路信息，以此完成对网络以区域为粒度的拓扑连接状态的详细描述。

$$\boldsymbol{A}[i][j]=\begin{cases}1, \text{枢纽域}u_i\text{与枢纽域}u_j\text{邻接}\\ 0, \text{枢纽域}u_i\text{与枢纽域}u_j\text{不邻接}\end{cases} \tag{1}$$

基于主动故障恢复思想，不相邻枢纽域间也需要路径规划，所以应提前明确这类枢纽域间的连接方式。由于枢纽域间的连接是双向的，因此只需遍历枢纽域邻接矩阵 $\boldsymbol{A}$ 的上三角元素。若 $\boldsymbol{A}[i][j]=0$，则进行枢纽域 u_i 与枢纽域 u_j 之间的连接计算，其算法如算法 3 所示，其中 hs 表示枢纽域 u_i， hd 表示枢纽域 u_j， root 表示为根节点。

算法 3　非邻接区域连接算法

```
1) root ← hs
2) builetree( node )
3)   root ← node
4)   if  root == hd  or  root.child 除父节点区域为空
5)     return
6)   end if
7)   root.child ← root 除祖先节点外的其他接节点
8)   for  node  in  root.child
9)     builetree( node )
10)  end for
```

算法 3 将枢纽域 u_i 作为树的根节点，并将其邻接的其他枢纽域作为它的子节点，遍历这些子节点，将子节点作为子树的根节点重复上述操作，直至遇到枢纽域 u_j 或者只有祖先节点的网络节点，则结束该次遍历并进入下一次遍历。当完成全部遍历，树的根节点到终端节点为 u_j 的路径即枢纽域 u_i 到枢纽域 u_j 的连接路径，这些路径可能存在多条，并且长短不一。

2.3　基于强化学习的子拓扑网络规划算法

实现枢纽域划分以及明确各个枢纽域之间的连接方式为规划子拓扑网络创造了条件。但由于网络拓扑中各链路剩余带宽随时间动态变化。为应对网络性能的动态变化特性，避免因为网络变化影响收敛路径质量，引入强化学习技术，利用强化学习自我探索环境的优势来应对网络环境的动态性变化，实现网络的高效收敛。

2.3.1　基于网络特征的智能体行为策略选择

强化学习中智能体从环境状态到行为动作形成映射关系，使累计的奖励值达到最大。agent 即强化学习中的智能体，在路由规划中，agent 从路由系统中接收当前状态信息和奖励信息，agent 选择的动作是路由系统从智能体接收到的输入。agent 在整个路由规划系统中，必须学习到最优的动作来使自身累计的奖励值达到最大，此时 agent 选择的动作即流量的最优路径。

agent 的任务是不断在系统中尝试从而学得一个策略[17-18]。使用 Q-Learning 算法通过判断相应状态下可选动作的 Q 值大小判别 agent 选择策略的好坏。Q-Learning 选择动作的行为称为策略 π。策略 π 根据状态 s 选择动作 a，可由式(2)表示。

$$a=\pi(s) \tag{2}$$

当 Q 值表达到收敛，策略 π 选择当前状态下最大 Q 值对应的动作来完成从初始状态到目标状态的转移。agent 具有探索和利用 2 种选择动作的方式。探索即选择之前未选择过的路径，从而寻找更多的可能性；利用即选择目前已选择过的路径，从而对已知的规划线路进行完善。如何选择策略来平衡探索环境和利用经验之间的关系，会对 Q 值的收敛带来严重影响。

ε-贪心策略（ε-greedy policy）基于探索因子 $\varepsilon(\varepsilon\in[0,1])$ 来平衡探索和利用。算法生成随机数 $\sigma(\sigma\in[0,1])$，当 $\sigma\leqslant\varepsilon$ 时，agent 使用随机策略，通过随机选取动作探索环境以获取经验[19]；当 $\sigma>\varepsilon$ 时，agent 使用贪婪策略利用已获得的经验[20]。ε-贪心策略的表达式为

$$\pi(s)=\begin{cases}a\sim A(s), \sigma\leqslant\varepsilon\\ \arg\max\limits_{a} \mathrm{Q}(s,a), \sigma>\varepsilon\end{cases} \tag{3}$$

通过调整探索因子 ε 的大小，可以影响 agent 是探索还是利用环境的倾向。为保证收敛速度，随着训练时间的增加而减小探索因子，即在训练早期更多地探索环境，到训练后期更多地利用环境，使 Q 值快速收敛到对应的解之中。但是 ε-贪心策略也存在弊端，

其无法在探索和利用之间达到一个较好的平衡。

在路径规划领域，agent 探索的环境具有网络特征，如链路长度、链路带宽、时延、跳数等。根据这些网络特征，促进 agent 获得有效探索，解决探索和利用的矛盾问题。上述的网络特征，以链路长度为例，如图 2(a)中节点 6 所示，其作为单源点，使用最短路径算法计算其他节点到该单源点的最短路径，最终将图 2(a)转换为图 2(b)。

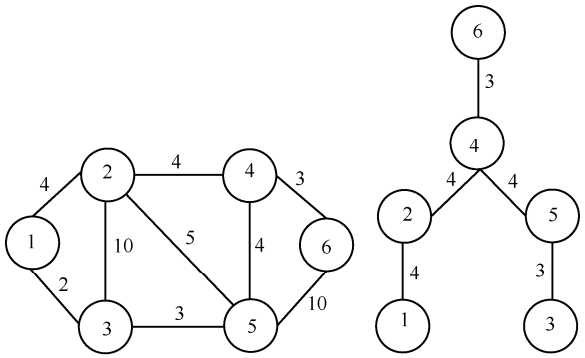

图 2 不同链路权值的拓扑及树状结构

从图 2(b)可得源节点 6 到任意节点的最短路径。将深度（depth）定义为节点 n 到源节点 source 路径中各链路长度之和，即

$$\text{depth}_n = \sum \text{depth}_n = \sum_{i=n}^{\text{source}} l_{(i,i+1)} \tag{4}$$

其中，$l_{(i,i+1)}$ 表示路径中第 i 个节点与第 $i+1$ 个节点间链路的长度。因为链路长度 l 为正值，则在一条路径 $\text{path} < n_0, n_1, \cdots, n_{\text{source}} >$ 中存在的关系为

$$\text{depth}_{n_i} \geqslant \text{depth}_{n_j}, i < j < \text{source} \tag{5}$$

使用强化学习实现最短路径规划，agent 的探索趋势是寻找深度越来越小的节点。

在网络中，除了链路长度、链路带宽、时延、跳数等可作为网络特征，各种特征通过权值加成也能作为一个新的特征。基于网络拓扑的特征，可以引导 agent 从探索阶段的高随机行为转变成高效探索，以此使学习网络更快地达到收敛。

2.3.2 基于区域特征的强化学习训练

将强化学习中的状态 s 定义为网络拓扑中的节点，将动作 a 定义为基于状态 s 选择邻接节点的行为。因为 agent 在探索网络拓扑时选定动作是随机的，所以 agent 的探索路径可能形成一个环路或者 agent 在 2 个节点间反复振荡等，这些情况都会严重影响模型收敛。同时当拓扑中出现链路故障，进行路由收敛操作时，对于受影响的流重新进行路径规划需要规避故障链路。

给定一个训练模型，如图 3 所示，在模型中存在枢纽节点、边缘节点、枢纽域以及链路。使用区域划分所构建的网络特征来引导强化学习 agent 进行环境的探索和促进算法模型训练的快速收敛。

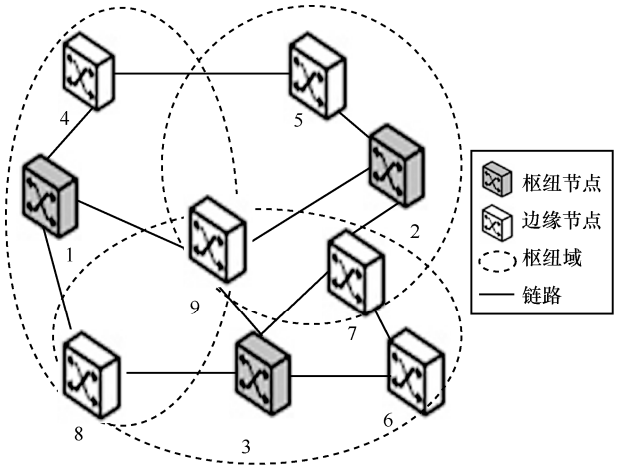

图 3 训练模型

根据定义 5 的规则，得到模型中环路路径每一步的 θ 情况。统计拓扑中一条路径的每一步 θ 值，计算得到一条环路中所有转移动作的 θ 累加值，即

$$\sum_{1}^{\text{len(path)}} \theta_{i,i+1} = 0 \tag{6}$$

而非环路路径对应的 θ 值之和与环路路径不同。通过随机选择 2 条非环路路径，计算 θ 值，即

$$\sum_{1}^{\text{len(路径1)}} \theta_{i,i+1} = 2 \tag{7}$$

$$\sum_{1}^{\text{len(路径2)}} \theta_{i,i+1} = 1 \tag{8}$$

在非环路路径中，路径方向因子的累加都为正值；而在 2 个节点中反复振荡的情形下，方向因子双向抵消。故而在非环路路径中当以源节点作为参考点规划出一条至目标节点时其 θ 累加值为正值的路径。

在强化学习 agent 探索环境时，agent 每一次选动作的行为就是选择邻接节点的过程，对应的效果正是远离源节点、靠近源节点或不靠近也不远离源节点。通过对比环路和非环路路径中的 θ 值集合，发现在环路 θ 值集合中出现更多的 θ 值为−1 的转移。因此为避免 agent 探索路径出现环路，在探索阶段使 agent 采取避免 θ 值为−1 的动作。

在 agent 探索阶段，如果每一步都选择 θ 值为非负的动作，可能使模型无法收敛到最优解甚至无法收敛。因为在路径规划时，会将时延、带宽等网络性能参数纳入计算。根据当下网络环境，从源节点到目的节点的路径可能出现绕路现象，但路径整体上的性能是优越的。此时虽然路径中整体的 θ 累加值为正值，但路径中部分转移的 θ 值可能存在负值。

在强化学习模型训练过程中，agent 从源节点开始通过探索和利用找到目的节点或者未找到目的节点，但 agent 状态转移次数达到设定值的过程称为一个 episode。而强化学习达到收敛需要迭代若干个 episode。为了避免网络收敛到次优解，Q-Learning 前期要进行充分的环境探索，所以在训练的初始阶段，允许 agent 的探索行为具有高度随机性。随着训练步数的递增，通过增大各链路网络特征的梯度差，agent 能实现高随机探索到高效探索的过渡，在提高收敛速度的同时保证能收敛到最优解。因此，在早期的 episode 中允许 agent 在探索阶段选择对应 θ 值为负的动作，而随着 episode 的不断迭代，减少 agent 探索对应 θ 值为负的动作的概率。以此保证在 agent 从环境中充分获得经验的同时提高探索效率，并减少在训练阶段环路的产生。

在该训练模型中，当选定拓扑中的网络节点被作为源节点时，其余节点根据所在枢纽域和源节点的位置关系得到如图 4 所示的拓扑分层结构。在图 4 中，当状态从上层节点转移到下层节点，体现为 θ 值非负的动作；当状态从下层节点转移到上层节点，体现为 θ 值为负值的动作。各分层之间的高度差即表示 θ 的绝对值大小。当高度差取向 0，表示各节点几乎在同一层，这时节点与节点之间的转移更少地受分层的影响，即表现为 agent 的随机探索状态。当不断提高分层之间的高度差时，引导 agent 向下层探索，即表现为 agent 的高效探索状态。

根据上述分析，设定函数为

$$f(t)=\begin{cases}1, t<\text{step}\\ (t-\text{step})^2, t\geqslant\text{step}\end{cases} \tag{9}$$

其中，函数 $f(t)$ 根据 episode 迭代进度对 θ 取绝对值；$h(\theta)$ 通过对应动作的状态决定 θ 的具体取值，如式(10)所示。

$$h(\theta)=f(\theta t) \tag{10}$$

此处设定了一个阈值 step，当 episode 迭代次数小于 step，则这些 episode 中所有 agent 在探索环境时，每一次的可选动作所对应的 θ 取值都为 1，这时 agent 采取了随机选择动作策略。当 episode 迭代到大于阈值 step 时，对于不靠近源节点的动作所对应的 θ 值会不断增大，而靠近源节点的动作所对应的 θ 值仍保持在 1 不变。随着 episode 的不断迭代，在可选动作中靠近源节点和不靠近源节点所对应的 θ 值之间的差值会越来越大。

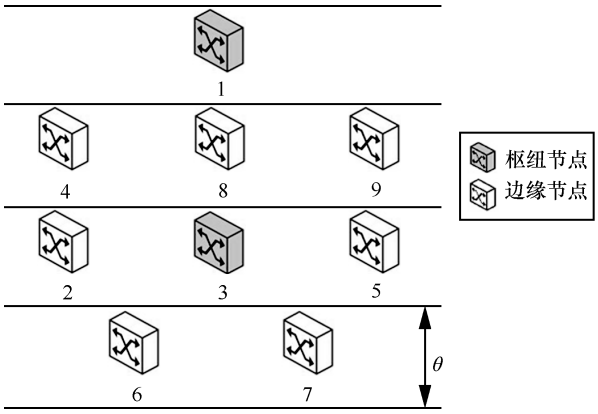

图 4　拓扑分层结构

定义当前状态下所有可选动作的 θ 值区间 D 如式(11)所示，区间取值范围为 0 到所有当前可选动作对应的 θ 值之和，区间划分成 n 份，n 为当前可选动作的数量，各子区间的长度为对应动作的 θ 值。通过在该区间上等概率取值，如式(12)所示，得到 η。再通过函数 $g(D,\eta)$，随机数 η 所在区间 D 对应的动作就是 agent 探索将要选择的动作。

$$D=\left[(0,\theta_1],(\theta_1,\theta_1+\theta_2],\cdots,\left(\sum_{i=1}^{n-1}\theta_i,\sum_{i=1}^{n}\theta_i\right]\right] \tag{11}$$

$$\eta=\text{random}(D) \tag{12}$$

$$g(D,\eta)=\begin{cases}1, 0\leqslant\eta\leqslant\theta_1\\ 2, \theta_1<\eta\leqslant\theta_1+\theta_2\\ \vdots\\ n, \sum_{i=1}^{n-1}\theta_i<\eta\leqslant\sum_{i=1}^{n}\theta_i\end{cases} \tag{13}$$

上述分析研究了强化学习中基于网络区域划分构建的特征来引导 agent 在探索阶段的行为。强化学习基于网络区域特征的策略如式(14)所示。

$$\pi_h(s)=\begin{cases}a\sim g(D,\eta), \sigma\leqslant\varepsilon\\ \arg\max\limits_a \text{Q}(s,a), \sigma>\varepsilon\end{cases} \tag{14}$$

通过 $h(\theta)$ 得到当前迭代 episode 内 agent 在探索阶段选择可选动作对应的 θ 值，将基于当前状态所对应的所有可选动作的 θ 值构建区间 D。随后通过在区间内等概率取值，由 $g(D,\eta)$ 获得对应区间的动

作，以此完成探索阶段 agent 选择动作的行为。

可见，在将网络进行基于枢纽节点的区域划分后，以枢纽域为单位进行粗粒度的路径规划，通过规避故障所在枢纽域和感知各枢纽域间的连通关系，区域划分可以引导 agent 在探索阶段的行为，让其高度的随机行为变得更有方向性，以此来加快强化学习的收敛，最终减少路径计算时间。

2.3.3 *算法实现*

利用网络区域构建的网络特征引导强化学习的探索行为，进而提出基于区域特征的行为策略。使用该策略作为强化学习模型中的行为策略以提高 agent 在探索阶段的效率，促进模型收敛。强化学习通过来自环境的奖励信号进行学习，为提高收敛路径的质量，将链路可用带宽、链路时延等指标引入奖励生成中，通过综合性能的奖励反馈，促使模型规划出高性能的路径。奖励反馈通过奖励函数 R 生成，当 agent 发生状态转移，将当前状态 s 和所选动作 a 输入函数 R，生成奖励来评价该状态转移。

一条链路的带宽由 2 个端口的能力决定，通过获取端口流量得到链路流量。OpenFlow 协议中提供机制来获取链路剩余带宽。控制器可通过周期下发 Port statistics 消息获得交换机端口的统计信息。从消息格式中发现可获取到收发的包数、字节数以及此次统计收发包数与字节数持续的时间。把 2 个不同时间的统计消息的字节数相减，再除以 2 个消息差即统计时间差就能得到统计流量速度。如果想得到剩余带宽，则使用端口最大带宽减去当前流量带宽，以此得到剩余带宽。

通过周期性地获取链路剩余带宽，设计奖励函数如式(15)所示，将剩余带宽作为正反馈，跳数作为负反馈。

$$R_t(s,a)=\alpha B-\beta t+\gamma\delta(s\_-d)-\delta \tag{15}$$

其中，R 为在节点 i 选择链路到节点 j 时所获得的奖励，α、β、γ、δ 为 4 个权衡四部分奖励权值的正值参数，B 为所选动作对应链路的剩余带宽，t 为对应链路时延，$\delta(j-d)$ 为激励函数，$s\_$ 为基于状态 s 在选择动作 a 后所转移的状态。如果 $s\_$ 为目的节点，则表示 agent 完成一次 episode 的训练，给予 1 的奖励。通过此类设置鼓励 agent 寻找目标。当 agent 每走一步给−1 奖励，用来控制源节点与目的节点间路径的长度。

为了保障当发生链路故障时，网络能迅速地完成故障恢复，降低故障影响，本节提出的基于强化学习的子拓扑网络规划算法仍是一种基于主动式故障恢复的思想。同时为保证网络收敛时的路由质量，设置了强化学习模型的周期性训练。当一个网络周期结束，SDN 控制器通过重新获取当前网络的链路可用带宽、链路时延参数来构建新的强化学习环境。强化学习模型基于新的环境进行训练，规划出新的子拓扑网络。在该周期内发生的链路故障由该子拓扑网络负责网络恢复操作，并在完成故障恢复后，使网络进入新的周期，具体算法如算法 4 所示。其中，$\boldsymbol{A}$ 为区域连接矩阵，hub 为枢纽节点集合，$\boldsymbol{T}_B$ 为网络链路剩余带宽矩阵，$\boldsymbol{T}_D$ 为网络链路时延矩阵，episode 为迭代次数，time 为单次最大训练次数，$g(v,e)$ 为初始化子网络，$\pi_h(s)$ 为基于区域特征的行为策略。

算法 4 基于强化学习的子拓扑网络规划算法

1)遍历 $\boldsymbol{A}$ 的上三角元素
2)if $\boldsymbol{A}[i][j]==1$
3) 源节点←hub[i]，目的节点←hub[j]
4) for i in range (episode)
5) $s\leftarrow$ 源节点，$d\leftarrow$ 目的节点
6) for j in range(time)
7) $a\leftarrow\pi_h(s)$
8) $s'\leftarrow a$
9) $r\leftarrow\alpha\boldsymbol{T}_B[s][s']-\beta\boldsymbol{T}_D[s][s']+\gamma\delta(s\_-d)-\delta$
10) 使用 Q 值函数更新 Q(s,a)
11) $s=s'$
12) if $s==d$
13) break
14) end if
15) end for
16) 使用 Q 值表计算 path(s,d)
17) $g^v\leftarrow$path(s,d)中的节点
18) $g^e\leftarrow$path(s,d)中的链路
19) end for
20)end if

通过算法 4 可以实现子拓扑网络的构建，这为实现高效的网络收敛提供支持。

3 仿真结果与分析

3.1 强化学习模型收敛测量

本文实验系统环境为 Ubuntu 16.04，采用 Ryu

控制器与 Mininet 对 QL-STCT 方案的有效性进行测试。在强化学习模型中，agent 探索的环境是 SDN 控制器通过集中化控制从数据转发平面获取网络全局视图所构建的。将 agent 探索环境的情景拟合成数据包在网络拓扑中转发的情景并定义强化学习的状态和动作。

状态。在网络拓扑中数据包的空间位置定义为状态，即一个交换机对应一个状态。本文中状态集合就是交换机集合，即 $S=[s_1,s_2,s_3,\cdots,s_{16}]$，其中 $s_1 \sim s_{16}$ 表示本文实验拓扑中的 16 台 OpenFlow 交换机，如图 5 所示。

动作。在网络拓扑中，数据包从一个交换机转发到另一个交换机的过程定义为强化学习中的动作。交换机只能将数据包发送到它的邻接交换机，强化学习模型中的状态对应的动作集合如式(16)所示。

$$\boldsymbol{A}(a \mid s_i)=\left\{\left(s_i \rightarrow s_j\right) \mid \boldsymbol{T}[i][j]=1\right\} \tag{16}$$

其中，$\boldsymbol{T}$ 为本实验中网络拓扑的连接矩阵，如式(17)所示。

$$\boldsymbol{T}[i][j]=\begin{cases}0, s_i, s_j \text{不相连} \\ 1, s_i, s_j \text{相连}\end{cases} \tag{17}$$

强化学习规划路径时，能够综合多个网络性能参数纳入考量，具体通过设置奖励函数来实现。在式(15)中，将链路可用带宽、链路时延作为规划路径的重要考虑参数。在强化学习模型迭代训练中，对于每一次的 episode，为了鼓励 agent 找到目的节点并控制 agent 探索路径的长度，当找到目的节点则给予正向反馈，当选择动作对应的节点不是目的节点则给予负反馈。因为侧重不同，所以为 4 个参数设定不同的权值。由于规划备用路径时关注链路带宽性能和链路时延，设置 α=0.4，β=0.3，γ=0.1，δ=0.2。

由强化学习算法 Q-Learning 算法可知，Q 值更新受到学习率 α 和折扣率 γ 影响。学习率 α 越小，保留训练的效果就越多，但模型的收敛速度将会越慢，且模型可能会出现过拟合；反之，学习率 α 越大，则训练的效果保留的就越少，但当学习率 α 过大时，模型容易出现振荡现象。折扣率 γ 越大，则未来奖励衰减越小，表明 agent 更关心长期奖励，反之表明 agent 更关心短期奖励。当折扣率 γ=0，则模型将学习不到未来的任何奖励信息，只关注当前状态的奖励；当折扣率 $\gamma \geqslant 1$，则奖励会因为不断累加且没有衰减造成算法无法收敛。故 α、γ 取值范围都为(0,1)。本文将强化学习模型中更新函数的学习率 α 设置为 0.9，折扣率 γ 设置为 0.8。

在 Q-Learning 中，算法收敛就是 Q 现实和 Q 估计逐渐靠近的过程。通过 Q 值更新函数中每次更新所有状态的误差和的计算体现强化学习的收敛

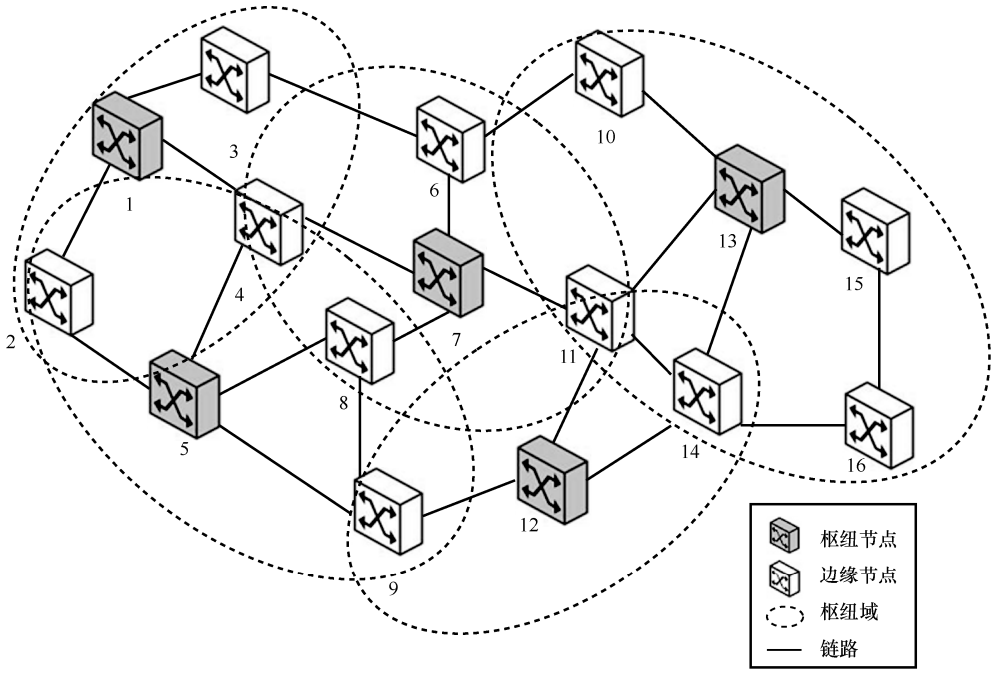

图 5　实验拓扑

过程。在 agent 每次做出决策选择动作进行状态转移时，Q 值函数评价该次行为策略，产生的误差值如式(18)所示。

$$\sigma = r_{t+1} + \gamma \max_{a} \mathrm{Q}(s_{t+1}, a) - \mathrm{Q}(s, a_t) \tag{18}$$

通过对一次 episode 中所有的误差值累加，如式(19)所示，e_i 大小可以定量地反映第 i 次 episode 训练情况。随着迭代次数的增加，e_i 趋于 0，表示每次迭代训练中 agent 的策略变得稳定，即 Q 现实与 Q 估计趋于同一值，算法达到收敛状态。

$$e_i = \sum_{j=0}^{\text{episode}} \sigma_j \tag{19}$$

本文实验通过统计算法训练过程中的 e_i，呈现算法的收敛过程。将基于网络区域特征的行为策略运用于强化学习模型训练中，并与使用 ε-贪心策略的模型训练进行对比。实验结果如图 6 所示。

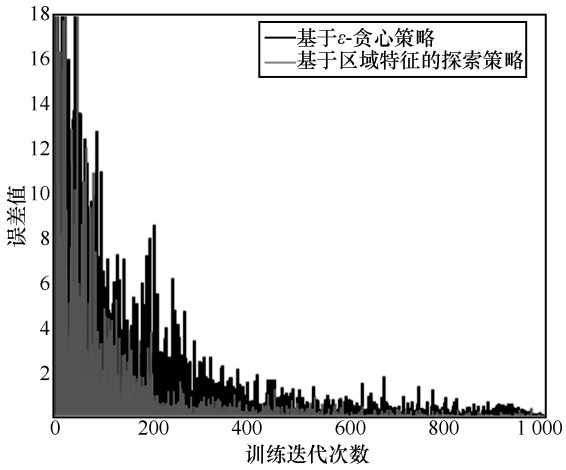

图 6 强化学习收敛过程

实验结果展示算法前 1 000 次训练的误差值，由此表明使用网络区域特征的行为策略运用于强化学习模型训练和使用 ε-贪心策略算法都具有不错的收敛能力。由图 6 可知，基于网络区域特征的行为策略算法能更快地实现收敛，且收敛后算法的波动值小于 ε-贪心策略。

3.2 路由收敛时间测量

本文实验通过查询服务器端主机的传输性能报告测量链路故障恢复时间，实验过程中运用了 3 种不同的路由收敛方案进行比较。方案 1 为 QL-STCT 方法；方案 2 为快速重路由技术（FRT, fast re-routing technique）方案[21]；方案 3 为只运用子拓扑网络规划算法而没有集成强化学习技术（STCT）方案。在出现丢包现象的时间间隔内，设置传输时间间隔为 1 s，统计丢包数量 loss 和发送数据包数量 total，通过式(20)计算丢包数量和发送数据包总量的占比。

$$T = \frac{\text{loss}}{\text{total}} \times 1\,000\ (\text{ms}) \tag{20}$$

链路故障恢复时间测量如图 7 所示，通过引入强化学习模型，提高了网络整体故障恢复的性能。由图 7 可知，使用 QL-STCT 方法进行故障恢复操作，共测量 50 次故障恢复时间，平均恢复时延为 21.6 ms，相较于 STCT 方案的 27.22 ms 的平均恢复时延，在时延指标上具有进一步的优化；相较于 FRT 方案更具有明显的优势。产生该结果是因为强化学习模块提高了备用路径的质量，进而缩短了整体的故障恢复时间。

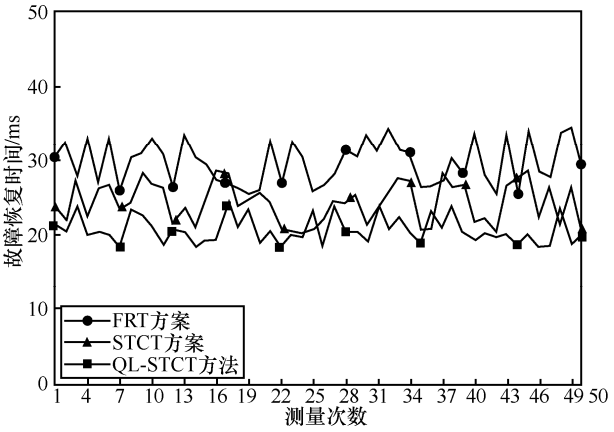

图 7 链路故障恢复时间测量

当多条链路发生故障时，QL-STCT 方法可同时进行多流的收敛操作，由于采用了构建子拓扑网络的方式进行路由收敛，链路故障恢复时间会保持在一个较稳定的范围内：链路故障不影响子拓扑网络链路时，受影响的网络节点都会将数据包转发给对应枢纽域内的枢纽节点，最终完成数据流的转发；当链路故障影响到子拓扑网络的链路时，QL-STCT 方法会重新构建子拓扑网络，以确保子拓扑网络性能。

3.3 网络带宽利用率测量

为验证 QL-STCT 方法提高链路带宽利用率的可行性，本文实验仍利用 Iperf 测量客户端主机指定的发送带宽和服务器端主机实际接收的带宽大小。当网络性能无法满足数据流的转发要求，网络出现拥塞和丢包现象时，服务器端接收到的数据量少于客户机端发送的数据量。此处定义流的带宽利

用率为服务端主机接收带宽与客户端主机发送带宽的比值，则网络的平均带宽利用率表达式为

$$\text{rate}=\frac{1}{n}\sum_{i=1}^{n}\frac{B_i^r}{B_i^s} \tag{21}$$

其中，n 表示网络中数据流总条数，B_i^s 表示数据流 i 对应客户端主机发送带宽大小，B_i^r 表示数据流 i 对应服务端主机接收的带宽大小。通过对网络中所有流的带宽利用率取平均值，得到网络的平均带宽利用率 rate 。

实验中为测试故障恢复后的网络性能，使用 Iperf 测量源主机的流量发送速率以及对应的目标主机的接收速率。依次增大源主机的流量发送速率，从 0.5 Mbit/s 到 5 Mbit/s 共测试 10 组数据，计算网络的平均带宽利用率，实验结果如图 8 所示。由图 8 可知，随着主机流量发送速率的不断增大，3 种方案中的网络平均带宽利用率均出现不同程度的下降。但整体而言 QL-STCT 方法具有更好的表现。当发送速率达到 5 Mbit/s 时，QL-STCT 方法的网络平均带宽利用率是 FRT 方案的近 4 倍。实验表明，基于 SDN 链路故障智能路由收敛方法能够很好地保障备用路径的性能。由于 QL-STCT 方法通过设置子拓扑网络将备用路径进行聚合来减少流表空间的消耗，对现实网络的影响较小，即便是更大的网络流量环境也具有很好的适应性。

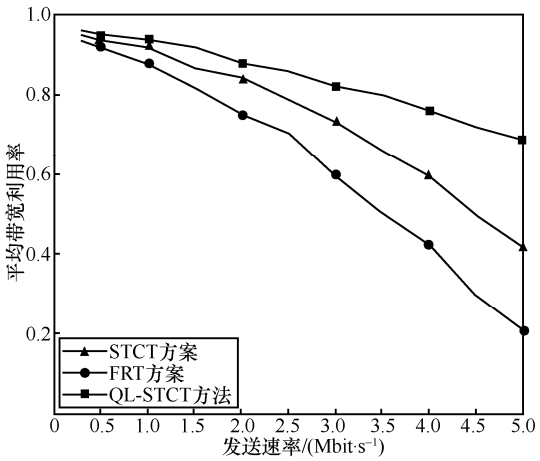

图 8　不同发送速率下网络平均带宽利用率

4　结束语

结合强化学习和子拓扑网络，并利用强化学习自我探索环境的特性动态规划子拓扑网络，保障备用路径性能，提出了一种 SDN 链路故障智能路由收敛方法。首先使用枢纽节点竞选算法筛选拓扑结构中的枢纽节点，根据枢纽节点进行区域划分，介绍基于网络特征的强化学习行为策略；然后将该策略用于强化学习模型训练，介绍基于强化学习的链路故障恢复算法；最后通过在仿真实验中模拟链路故障恢复来检验该算法的有效性。仿真结果表明，所提出的路由收敛算法通过周期性的训练强化学习模型，规划备用路径，保障备用路径的性能，有效地提升了网络收敛速度，保障了网络在出现故障恢复后的整体性能。后续工作将在子拓扑网络规划算法中进一步研究引入深度强化学习，以实现更大网络规模链路故障发生时高效的智能路由收敛，以改善由于 Q 值表增大而导致数据查找和存储带来的资源消耗等问题，提升算法性能。

参考文献：

[1] 邓书华, 卢泽斌, 罗成程, 等. SDN 研究简述[J]. 计算机应用研究, 2014, 31(11): 3208-3213.
DENG S H, LU Z B, LUO C C, et al. Outline of software defined networking[J]. Application Research of Computers, 2014, 31(11): 3208-3213.

[2] ABDALLAH S, KAYSSI A, ELHAJJ I H, et al. Network convergence in SDN versus OSPF networks[C]//Proceedings of 2018 Fifth International Conference on Software Defined Systems (SDS). Piscataway: IEEE Press, 2018: 130-137.

[3] ABDALLAH S, KAYSSI A, ELHAJJ I H, et al. Performance analysis of SDN vs OSPF in diverse network environments[J]. Concurrency and Computation: Practice and Experience, 2020: doi.org/10.1002/cpe.5410.

[4] NOTO M, SATO H. A method for the shortest path search by extended Dijkstra algorithm[C]//Proceedings of 2000 IEEE International Conference on Systems, Man and Cybernetics. Piscataway: IEEE Press, 2000: 2316-2320.

[5] 龙昭华, 叶二伟, 董瑞芳. SDN 中基于多指标的链路负载均衡模型[J]. 计算机工程与设计, 2019, 40(4): 948-952, 1084.
LONG Z H, YE E W, DONG R F. Load balancing model based on multi-indexes in SDN[J]. Computer Engineering and Design, 2019, 40(4): 948-952, 1084.

[6] YAN J Y, ZHANG H L, SHUAI Q J, et al. HiQoS: an SDN-based multipath QoS solution[J]. China Communications, 2015, 12(5): 123-133.

[7] 池亚平, 高聪, 陈颖, 等. SDN 架构下的链路分离路径算法的研究[J]. 计算机应用与软件, 2018, 35(9): 183-188, 235.
CHI Y P, GAO C, CHEN Y, et al. Research on link disjointed path algorithm in SDN architecture[J]. Computer Applications and Software, 2018, 35(9): 183-188, 235.

[8] 周飞杰, 张坤丽, 王国卿, 等. SDN 中基于遗传机制的自适应路由算法研究[J]. 计算机工程与应用, 2019, 55(2): 86-91, 186.
ZHOU F J, ZHANG K L, WANG G Q, et al. Research of adaptive SDN routing algorithm based on genetic mechanism[J]. Computer Engineering and Applications, 2019, 55(2): 86-91, 186.

[9] 杨洋, 杨家海, 秦董洪. 数据中心网络多路径路由算法[J]. 清华大学学报(自然科学版), 2016, 56(3): 262-268.
YANG Y, YANG J H, QIN D H. Multipath routing algorithm for data center networks[J]. Journal of Tsinghua University (Science and Technology), 2016, 56(3): 262-268.

[10] 农黄武, 黄传河, 黄晓鹏. 基于 SDN 的胖树数据中心网络的多路径路由算法[J]. 计算机科学, 2016, 43(6): 32-34, 76.
NONG H W, HUANG C H, HUANG X P. SDN-based multipath routing algorithm for fat-tree data center networks[J]. Computer Science, 2016, 43(6): 32-34, 76.

[11] ARINEITWE F, SERUGUNDA J, OKELLO D. Development of the protocol for inter-autonomous systems routing in software defined networks[C]//2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC). Piscataway: IEEE Press, 2021: 414-422.

[12] FU Q X, SUN E C, MENG K, et al. Deep Q-learning for routing schemes in SDN-based data center networks[J]. IEEE Access, 2020, 8: 103491-103499.

[13] HAN Q W, CHENG S, ZENG L Q. An intellectual routing algorithm based on SDN[C]//Proceedings of 2020 IEEE/CIC International Conference on Communications in China (ICCC). Piscataway: IEEE Press, 2020: 1144-1149.

[14] 程成. SDN 下基于强化学习的路由规划算法研究[D]. 杭州: 浙江工商大学, 2018.
CHENG C. Research on routing algorithm based on reinforcement learning in SDN[D]. Hangzhou: Zhejiang Gongshang University, 2018.

[15] CHEN Y R, REZAPOUR A, TZENG W G, et al. RL-routing: an SDN routing algorithm based on deep reinforcement learning[J]. IEEE Transactions on Network Science and Engineering, 2020, 7(4): 3185-3199.

[16] 任晓龙, 吕琳媛. 网络重要节点排序方法综述[J]. 科学通报, 2014, 59(13): 1175-1197.
REN X L, LYU L Y. Review of ranking nodes in complex networks[J]. Chinese Science Bulletin, 2014, 59(13): 1175-1197.

[17] 高阳, 陈世福, 陆鑫. 强化学习研究综述[J]. 自动化学报, 2004, 30(1): 86-100.
GAO Y, CHEN S F, LU X. Research on reinforcement learning technology: a review[J]. Acta Automatica Sinica, 2004, 30(1): 86-100.

[18] YU H Z, BERTSEKAS D P. On boundedness of Q-learning iterates for stochastic shortest path problems[J]. Mathematics of Operations Research, 2013, 38(2): 209-227.

[19] 陈建平, 邹锋, 刘全, 等. 一种基于生成对抗网络的强化学习算法[J]. 计算机科学, 2019, 46(10): 265-272.
CHEN J P, ZOU F, LIU Q, et al. Reinforcement learning algorithm based on generative adversarial networks[J]. Computer Science, 2019, 46(10): 265-272.

[20] 檀朝东, 蔡振华, 邓涵文, 等. 基于强化学习的煤层气井螺杆泵排采参数智能决策[J]. 石油钻采工艺, 2020, 42(1): 62-69.
TAN C D, CAI Z H, DENG H W, et al. Intelligent decision making on PCP production parameters of CBM wells based on reinforcement learning[J]. Oil Drilling & Production Technology, 2020, 42(1): 62-69.

[21] MUTHUMANIKANDAN V, VALLIYAMMAI C. Link failure recovery using shortest path fast rerouting technique in SDN[J]. Wireless Personal Communications, 2017, 97(2): 2475-2495.

[作者简介]

李传煌（1980– ），男，江西九江人，博士，浙江工商大学教授、硕士生导师，主要研究方向为软件定义网络、开放可编程网络、边缘计算、人工智能应用。

陈泱婷（1998– ），女，浙江绍兴人，浙江工商大学硕士生，主要研究方向为软件定义网络、人工智能应用等。

唐晶晶（1998– ），女，浙江杭州人，浙江工商大学硕士生，主要研究方向为软件定义网络、人工智能应用等。

楼佳丽（1998– ），女，浙江东阳人，浙江工商大学硕士生，主要研究方向为软件定义网络、人工智能应用等。

谢仁华（1997– ），男，浙江临海人，浙江工商大学硕士生，主要研究方向为软件定义网络、人工智能应用等。

方春涛（1995– ），男，浙江建德人，浙江工商大学硕士生，主要研究方向为软件定义网络、人工智能应用等。

王伟明（1964– ），男，浙江遂昌人，博士，浙江工商大学教授、硕士生导师，主要研究方向为新一代网络架构、开放可编程网络。

陈超（1986– ），男，浙江湖州人，博士，浙江工商大学副教授、硕士生导师，主要研究方向为下一代无线通信网络技术、网络编码、机器/深度学习等。