doi:10.11887/j.issn.1001-2486.25050027 http://journal.nudt.edu.cn

深度强化学习和负载中心性理论融合的分段路由优化算法

曹继军[1],吴宗明[2],汤 强[2*],李小玉[3]

(1.国防科技大学 计算机学院,湖南 长沙 410073;2.长沙理工大学 计算机学院,湖南 长沙 410114;
3.云南大学 信息科学与工程学院,云南 昆明 650500)

摘 要:结合软件定义网络与分段路由(segment routing,SR)可优化网络性能,但在大规模动态网络中,其关键节点链路利用率过高会导致队列延迟激增。为此,提出深度强化学习与负载中心性理论融合的分段路由优化算法(segment routing optimization algorithm fusing deep reinforcement learning and load centrality theory,SROD-LC)。通过负载中心性理论量化网络节点重要性,识别关键节点并监控其链路负载状态;利用多智能体强化学习框架,在关键节点部署分布式深度强化学习智能体,通过共享奖励机制协调路由决策,实现链路负载的主动优化。同时结合SR的灵活性,动态调整段标识列表快速重路由部分流量,降低本地链路利用率并规避潜在拥塞。基于真实网络拓扑的模拟实验结果表明:当SR关键节点比例在0.3~0.5范围时,SROD-LC优化效果显著,与基准算法相比,可将网络最大链路利用率降低21%~35%。

关键词:深度强化学习;分段路由;流量工程;软件定义网络;负载中心性

中图分类号:TP393 **文献标志码**:A **文章编号**:1001-2486(2025)06-046-14

论文拓展

Segment routing optimization algorithm fusing deep reinforcement learning and load centrality theory

CAO Jijun[1], *WU Zongming*[2], *TANG Qiang*[2*], *LI Xiaoyu*[3]

(1. College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China;
2. School of Computer, Changsha University of Science and Technology, Changsha 410114, China;
3. School of Information Science and Engineering, Yunnan University, Kunming 650500, China)

Abstract: Combining software defined networking and SR (segment routing) can optimize network performance, but in large-scale dynamic networks, excessive link utilization at key nodes can lead to a surge in queue delays. To address this, a SROD-LC (segment routing optimization algorithm based on deep reinforcement learning and load centrality theory) was proposed. By quantifying the importance of network nodes using load centrality theory, key nodes are identified and their link load states are monitored; utilizing a multi-agent reinforcement learning framework, distributed deep reinforcement learning agents are deployed at key nodes, coordinating routing decisions through a shared reward mechanism to achieve proactive optimization of link loads. At the same time, leveraging the flexibility of SR, segment identifier lists are dynamically adjusted to quickly reroute partial traffic, reducing local link utilization and avoiding potential congestion. Simulation experiments based on real network topologies show that when the proportion of SR key nodes is in the range of 0.3~0.5, the SROD-LC algorithm exhibits significant optimization effects, reducing the network's maximum link utilization by 21%~35% compared to baseline algorithms.

Keywords: deep reinforcement learning; segment routing; traffic engineering; software defined networking; load centrality

作为网络优化的关键技术,流量工程[1](traffic engineering,TE)已成为学术界和工业界关注的热点领域。近年来,随着网络运营商逐步引入分段路由(segment routing,SR)与软件定义网络(software defined networking,SDN)相结合的技术[2],SR路由问题已成为研究热点。尽管SDN

收稿日期:2025-06-16

基金项目:国家自然科学基金资助项目(62272063);湖南省教育厅科研基金资助项目(23A0258);湖南省自然科学基金资助项目(2021JJ30736,2023JJ50331);长沙市自然科学基金资助项目(kq2014112)

第一作者:曹继军(1979—),男,陕西汉中人,副研究员,博士,硕士生导师,E-mail:caojijun@nudt.edu.cn

***通信作者**:汤强(1982—),男,湖南长沙人,讲师,博士,硕士生导师,E-mail:tangqiang@csust.edu.cn

引用格式:曹继军,吴宗明,汤强,等. 深度强化学习和负载中心性理论融合的分段路由优化算法[J]. 国防科技大学学报,2025,47(6):46-59.

Citation: CAO J J, WU Z M, TANG Q, et al. Segment routing optimization algorithm fusing deep reinforcement learning and load centrality theory[J]. Journal of National University of Defense Technology, 2025, 47(6): 46-59.

和 SR 技术可以独立应用,但两者的结合能够实现更为优化的路由解决方案。具体而言,基于分布式控制平面的 SR 技术,能够在网络故障时实现亚秒级备用路径切换。然而,结合 SDN 集中控制与 SR 技术,不仅能够支持更广泛的应用场景,而且可以使网络运营商充分挖掘 SR 源路由技术的潜力。通过集中式 SDN 控制器,可利用网络全局视图收集和监控拓扑变化,借助标准的路径计算单元(path computation element, PCE)计算框架,通过路径计算单元通信协议(path computation element communication protocol, PCEP)等标准协议[3]对入口节点进行路径信息更新,从而实现对网络拓扑变化的快速响应。因此,SR 与 SDN 相结合成为流量工程领域网络路由技术研究的重点方向。

高度动态的通信网络对网络管理提出严峻挑战,超可靠、低延迟通信场景对端到端延迟和可靠性要求更高。传统基于 TE 的 SR 路由优化方案通过优化路径长度实现路由优化,并支持节点或链路故障时的快速恢复[4]。然而,此类方案[5-6]在动态网络环境中响应速度不足,可能导致关键节点链路利用率过高,增加负载,引发流量高峰期网络拥塞,影响性能。集中式拥塞控制方案[7-8]虽能缓解局部拥塞,但因可扩展性不足,通常以分钟级频率检测拥塞,当关键节点拥塞时,网络性能已显著下降。SDN 与 SR 结合的路由方案在路径计算、流表下发及动态调整(如重路由、故障恢复)方面面临响应速度挑战[9]。在动态网络中,毫秒级延迟要求使得应对关键节点链路过载成为难题。新的路由方案需自主监测并提前发现过载风险,通过重路由绕开高负载节点或链路。

基于深度强化学习(deep reinforcement learning, DRL)的路由解决方案旨在重路由关键节点的部分流量,将关键节点的链路负载控制在预先设定的阈值以下,从而提升网络整体性能。为了实现该目标,SR 技术通过重定义段标识(segment identifier, SID)列表,可快速调整路由路径,绕过高时延节点或利用率过高的链路。因此,路由更改借助 SR 技术的灵活性得以实现:路由器可在本地执行无循环重新路由操作,具体方式为在 SR 报文头中插入新的编码路径,即执行 SR 封装,从而生成最优 SR 路径。进一步地,本文在多个关键节点中部署 DRL 智能体。通过定义临时奖励函数,这些智能体能够以协调的方式进行训练,确保不同智能体的路由决策一致性及优化效果的实现。

本文还提出基于关键节点状态检测和流量控制的优化方案,通过实时监测链路负载和队列占用率,识别接近拥塞状态,触发路由调整;提出 DRL 和负载中心性(load centrality, LC)理论[10]融合的分段路由优化算法(segment routing optimization algorithm based on deep reinforcement learning and load centrality theory, SROD-LC),在关键节点基于节点状态和网络信息动态调整流量分布,缓解拥塞,提升性能。本文的主要贡献包括:

1)将负载中心性理论引入分段路由优化中,用于量化网络中节点的重要性,识别和监控关键节点的负载状态。基于该理论,本文选择关键节点并部署智能体,利用 DRL 模型实现了对高负载链路的实时监控和动态调整。通过预测潜在拥塞风险,算法能够在链路过载前主动重路由部分流量。

2)设计了一种基于多智能体强化学习(multi-agent reinforcement learning, MARL)的分布式路由优化框架,在关键节点部署 DRL 智能体,通过共享奖励机制实现协作优化。该框架结合全局与局部奖励优化路由策略,协调智能体决策,避免路径冲突和负载过高。智能体利用局部和全局信息学习流量分布,自适应调整重路由策略,显著降低最大链路利用率(maximum link utilization, MLU)。此外,结合离线训练与在线部署,智能体能够快速决策,有效应对动态流量变化。

3)在真实通信网络拓扑场景下开展模拟实验,将提出的 SROD-LC 与现有基准算法进行对比。实验结果表明,SROD-LC 在链路利用率、路径冲突率和动态流量适应性等方面均表现出显著优势,能够有效提升网络性能。

1　相关工作

1.1　基于 SR-TE 的 SR 路由优化

当前关于分段路由流量工程(segment routing traffic engineering, SR-TE)的路由优化研究已取得显著进展[11]。Roelens 等[12]提出的 TI-LFA 解决方案已经在实际网络中成功部署,可实现快速重路由(fast re-route, FRR)功能。TI-LFA 方案通过预先计算重路由路径,在检测到故障后能够迅速激活备用路径。Lemeshko 等[13]提出了类似的方案,旨在应对单点故障(如节点或链路故障),同样可提供快速重路由能力。与文献[5-6]类似,Sheu 等[14]提出了一种基于 SDN 的 SR-TE 的路由优化方案,以避免拥塞并选择具有更多剩余带宽

资源的链路。借助集中式控制器的全局网络视图,上述方案能够根据最新收集的网络信息计算段列表,从而有效适应动态网络场景。在近期研究中,Brundiers 等[15-16]提出了一种基于中点的优化方法,用于计算缓解拥塞的备选 SR 策略。与文献[4-6]中针对链路故障的网络恢复以及集中式拥塞缓解方案不同,Medagliani 等[17]则提出了一种分布式拥塞管理(congestion management,CM)机制。CM 机制通过 SR 在非等价多路径(unequal cost multi paths, UCMP)上分配流量,实现拥塞链路的流量卸载。实验结果表明,该方案在缓解网络拥塞方面表现优异,取得了较好效果。高新成等[18]针对 SDN 流表低效及控制器高开销问题,提出混合分段路由概率流调度机制,周期区分流量,小流用等价多路径路由算法分配路径,大流用粒子群优化路径,通过概率及选择性调度降低控制器与设备间负载开销。尽管 SR-TE 路由优化方案成效显著,但在大型通信网络中仍面临响应速度和优化效果的限制。因此,研究者探索将智能优化技术引入 SR,提升自适应能力和性能,为基于强化学习的 SR 路由优化开辟新方向,下一节将详述相关进展。

1.2　基于强化学习的 SR 路由优化

随着 RL/DRL 技术的快速发展[19],其在 SR 网络路由优化中的应用取得了显著成果。Tian 等[20]提出了 WA-SRTE 的路由方案。该方案将 TE 问题转化为深度强化学习问题,并对 SRv6 节点部署和流量路径进行联合优化。实验结果表明,WA-SRTE 方案能够实现与完整 SR 网络几乎相当的性能表现。针对 SR 网络中静态 TE 策略的灵活性不足问题,Chen 等[21]提出了一种基于 DRL 的 TE 优化方案。该方案根据实时网络流量分布动态调整不同路径间的分流比例,有效缓解了网络拥塞问题,显著提升了网络性能。Ren 等[22]从流量工程的角度研究了 SRv6 增量部署(SRv6 incremental deployment,SRID)问题,旨在提供更均衡的网络服务。他们将 SRID 问题形式化为一个带整数规划的问题,并提出了两种解决方法:一种是考虑短期影响的强化学习方法,另一种是考虑长期影响的贪心方法。实验结果表明,两种解决方法均能有效降低最大链路利用率。

此外,Tong 等[23]提出了一种基于 SDN 和 SR 相结合的感知 SR 路由解决方案。该方案采用流量分类方法,并结合强化学习技术,以适应动态网络环境,从而优化网络性能。进一步,Aureli 等[24]提出了一种基于 DRL 的框架,用于主动维护 SR 节点的链路负载在预先设定的阈值范围内。该框架通过选择重新路由操作,将流量从过载链路转移到替代路径。在此基础上,Aureli 等[25]进一步提出本地网内链路负载控制的 SR 重路由(SRv6 rerouting for local in-network link load control,SR-LILLC)方案。与文献[5-6]集中式监测拥塞的方案不同,该方案在文献[24]的研究基础上进行了扩展,通过在节点子集中配备本地智能体,实现了 SRv6 重路由本地链路负载机制。该方案使端到端延迟的最高值减少了约 20%。此外,Wang 等[26]针对 SR-TE 面临的主要问题,包括计算时间长、控制开销高和部署成本高等,提出了基于智能选择节点(smart node selection,SNS)的 SR 路由方案。该方案通过选择节点子集作为候选中间节点进行流量路由,有效降低 SR 的开销和部署成本。赵鹏程等[27]针对 SRv6 网络关键流识别不足及动态环境适应性差的问题,提出深度强化学习算法识别关键流并重路由关键流,通过线性规划优化关键流路径,结合差异化路由实现普通流与关键流负载均衡。

现有研究为 SR 路由优化提出了多种创新方法,但在高度动态的通信网络中,快速响应以均衡关键节点链路利用率、缓解拥塞至关重要。此外,关键节点链路利用率过高易导致流量积压和负载加剧,该问题亟须解决。

2　SR 路由优化整体框架

SR 与 SDN 技术相结合在路由优化中展现出显著优势。在 SDN 架构中,控制器通过南向接口协议(如 OpenFlow)能够实时获取网络状态信息并下发路由转发规则[28]。SR 作为一种实现源路由范式的网络结构,支持在底层网络上构建端到端隧道,具体基于多协议标签交换(muti-protocol label switching,MPLS)或互联网通信协议第六版(Internet protocol version 6,IPv6)[29-30]实现。SR 技术引入 SDN 后,流量路由过程如图 1 所示。对于每条流,控制器只需与源节点交互,完成路径编码并下发相应的段列表。

2.1　基本思想

本文研究场景为互联网服务提供商(Internet service provider,ISP)运营的骨干网络,管理和控制平面的目标是通过优化流量调度提升网络资源利用率,限制链路利用率在预设阈值内,防止关键节点因流量突增而出现链路过载、影响服务质量(quality of service,QoS)。本文研究重点为骨干网

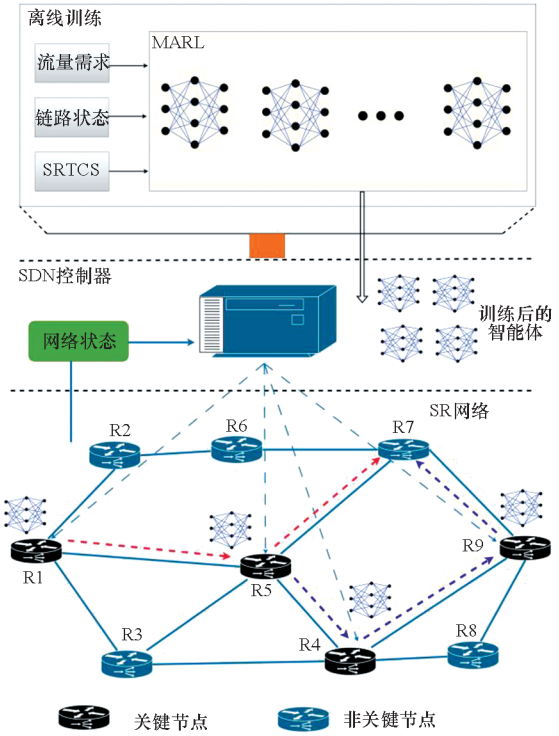


图 1　基于 SDN 的 SR 路由框架

Fig. 1　SDN-based SR routing framework

络关键节点链路利用率的优化问题。

为解决关键节点链路过载,本文提出本地主动重路由方案,通过实时监控关键节点传出链路负载实现有效控制。当负载超阈值时,系统启动重路由机制,调整部分流量路径。但关键节点无法感知其他节点的重路由操作,可能导致路径冲突,降低网络性能。为此,本文以最大链路利用率值 U_{MLU} 作为性能评估指标,U_{MLU} 增加表明性能下降,需优化路由策略。

具有 SR 功能的网络节点具有一项重要特性,即特定分段路由流量计数器[31](segment routing traffic counters,SRTCS)的可用性。该功能能够实现对网络流量的精确统计和细粒度测量,为网络性能评估提供重要依据。在 SR 网络设备中,与 SR 流量相关的统计信息被系统记录,这些数据可用于量化评估特定网段或路径的流量指标(如流量统计值、路径利用率等)。通过利用 SRTCS 功能,具备 SR 功能的节点能够实时采集流量统计数据,并基于活动网段对数据进行聚合处理。这种统计方式使得链路上的流量能够按活动网段进行区分,从而实现更为精细的流量统计,准确反映链路的实际负载情况。基于上述特性,本文在关键网络节点部署智能体,以执行更为精细化的控制平面路径计算(重路由)操作。在路径规划过程中,新路径需满足以下要求:有效降低本地链路过载风险、需确保全局路径的一致性、应避免因多个关键节点的重路由操作而导致链路过载问题的叠加。

2.2　SR 路由优化框架概述

本文提出一种关键节点链路负载控制的智能路由策略。该策略通过在关键节点执行更细粒度的路径计算,提升路由优化效果。具体而言,关键节点路由器能够感知本地网络状态信息,并基于此动态调整流量路径,以避免过载并提升服务质量。本地网络状态信息具有较高的时间分辨率,包括链路负载和流量计数器等指标。对于全局网络参数的部分信息(如远程链路的负载),这类信息可由 SDN 控制器获取并用于辅助决策。

为了实现关键节点链路负载控制的智能路由策略,本文采用深度强化学习的近端策略优化(proximal policy optimization,PPO)模型,用于学习网络参数、本地网络信息参数(高时间粒度)和全局网络信息参数(低时间粒度或缺失)之间的隐藏关系,从而准确识别需要执行的路径修改操作。新路径需在减少本地链路过载的同时,确保网络性能不受影响,即维持全局路径一致性并避免不同关键节点中的路径过载现象。此外,在路由优化方案中,本文将独立的 DRL 智能体部署于网络关键节点,并通过协作方式完成离线训练。训练完成后,每个智能体在关键节点上独立运行,依据本地状态信息自主选择 SR 本地重路由操作。通过这种方式,本文的 SR 路由解决方案能够实时跟踪本地流量动态变化,并在毫秒级时间尺度上做出响应。

图 1 展示了一个基于 MARL 的 SDN-SR 路由架构。假设一个由 9 个节点和多条链路组成的 SR 网络,其中多个智能体分别部署在关键节点路由器 R1、R5、R4 和 R9 上。智能体的训练在虚拟环境中以共享方式离线完成,训练后的模型被在线部署到关键节点中,能够在实际网络中实时操作。在图 1 中,全局网络状态由 SDN 控制器监控和收集,包括流量需求、链路状态和服务需求(如 SR-TE)。控制器将这些信息分发给智能体,智能体根据本地网络状态独立执行路径优化操作。初始路径由标准内部网关协议(interior gateway protocol,IGP)计算,采用最短路径策略。例如,从 R1 到 R7 的初始路径用红线表示。当链路 R5→R7 出现过载情况时,部署在 R5 的智能体可以通过修改 SID 列表重新规划路径(如紫线所示)。初始路径通过节点 R5 到达 R7,但新的路径可能选择通过 R4 和 R9 到达 R7,确保本地链路负载

不高于确定阈值以及 QoS 性能。

综上所述,本文提出的基于 DRL 和 SR 的路由优化方案,通过在关键节点部署智能体,能够实现网络性能的自主优化,并在链路负载过高的情况下主动采取行动,确保网络的高效运行。

2.3 网络关键节点选择

在通信网络中,具有高负载中心性的节点更容易成为拥塞热点。通过识别负载较高的节点,充分利用 DRL 智能体对这些节点进行控制以实现 SR 重路由,可有效降低拥塞风险,并将关键链路的链路利用率控制在阈值以下,从而确保整体网络的高效运行。高负载关键节点在通信网络中承担着大量数据包传输任务。负载中心性指标可用于识别网络中潜在的拥塞热点和高利用率节点,从而为优化网络流量分布和提升整体传输效率提供依据。为此,本文引入负载中心性指标,用于量化网络中每个节点传输负载的重要性和集中性。负载中心性是描述网络节点重要性的核心指标,在优化网络性能及设计更高效的网络系统方面具有重要的应用价值[32]。设 $G(V,E)$ 表示通信网络拓扑图,其中 V 为网络节点集合,E 为节点链路集合。对于任意一对节点$(s,\ d)$,网络中可能存在多条最小权重路径。令 $\theta_{s,d}$ 表示从节点 s 发送到节点 d 的流的数量。假设流总是按照最小权重路径传递到下一跳节点。如果有多个下一跳,流量将在它们之间平均分配。节点 v 转发的流的数量记为 $\theta_{s,d}(v)$,则节点 v 的负载中心性定义如下:

$$C_{\mathrm{LC}}(v) = \sum_{s,d\in v}\theta_{s,d}(v) \tag{1}$$

通常假设源节点 s 和目的节点 d 为不同节点。C_{LC} 由网络拓扑结构及其最小权重路径发现算法决定。对于无向图,节点对的数量为 $\frac{N(N-1)}{2}$。为了便于比较和分析,可对负载中心性进行归一化处理,归一化公式定义如下:

$$\overline{C_{\mathrm{LC}}} = \frac{2}{N(N-1)}\sum_{s,d\in v}\theta_{s,d}(v) \tag{2}$$

为了防止网络关键节点利用率过高导致的拥塞现象,本文通过在关键节点部署的智能体动态调整路由策略。然而,网络环境中的新流不断出现,同时旧的流可能会消失或发生变化,这导致关键节点需随之动态调整。如何动态选择关键节点以适应网络的变化成为需要解决的重要问题。

针对上述问题,采用基于周期性时间段和增量更新机制的负载中心性动态更新方法。通过定期计算和更新节点的负载中心性,可以实时观察智能体的性能表现。该方法能够及时响应流量变化,动态调整关键节点的选择,从而有效避免因流量波动导致的网络性能下降。对于网络中节点的负载中心性计算,采用基于时间段的更新机制。具体而言,节点的负载中心性值是基于过去时间段内的链路负载进行计算的,且该时间段会随着时间推移而动态调整。假设链路负载为 $y_{\mathrm{L}}(v,j)$ 和链路容量为 c_e,则在时间段 T 内,节点 v 的负载中心性 $C_{\mathrm{LC}}(v,T)$ 可以计算为:

$$C_{\mathrm{LC}}(v,T) = \sum_{s,d\in v}\frac{y_{\mathrm{L}}(s,d,T)}{c_e} \tag{3}$$

本文采用上述公式对节点的负载中心性进行周期性计算。负载中心性的计算结果会随着时间段 T 的更新而动态调整,以反映网络中最新的流量情况。

为了提高计算效率,引入增量更新机制。该机制允许系统仅在关键节点链路负载发生变化时对相关节点的负载中心性进行更新,而非在每个时段内都重新计算所有节点的负载中心性。通过这种机制,可以显著降低计算复杂度,尤其是在大规模网络中。具体而言,当链路负载 $y_{\mathrm{L}}(v,j)$ 发生变化时,系统仅需对节点 v 的负载中心性进行增量更新。假设节点 v 的当前负载中心性为 $C_{\mathrm{LC_{old}}}(v)$,链路负载变化为 $\Delta y(v,j)$,则更新后的负载中心性为:

$$C_{\mathrm{LC_{new}}}(v,T) = C_{\mathrm{LC_{old}}}(v) + \frac{\Delta y(v,j)}{c_e} \tag{4}$$

通过上述机制,系统仅需更针对变化的节点链路,而无须对所有链路重新计算负载中心性。然而,关键节点的过多变化可能导致线下训练的频繁变动,从而增加训练的计算开销。为此,引入关键节点变化阈值 μ,仅当关键节点的变化量(基于初始选择)超过该阈值时,才执行增量更新以及线下训练。这种机制可有效避免对微小波动的过度响应。本文将关键节点变化阈值设置为 $\mu=50\%$。

将周期时间段与增量更新机制结合可在确保系统实时响应的同时,显著减少计算开销。每个节点维护一个周期时间段来计算负载中心性,时间段内的负载数据会不断更新。当链路负载变化导致关键节点的变动超过阈值 μ 时,系统通过增量更新机制来重新计算负载中心性,从而避免每次都重新计算。以此动态选择那些负载较重、很可能成为网络性能瓶颈的节点进行流量重路由。在智能体训练过程中,周期时间段和增量更新机

制的动态调整可帮助智能体逐渐学习关键节点的信息,从而优化路由路径。

3　SROD-LC 路由优化算法

3.1　SROD-LC 路由优化问题

本文研究的路由优化问题的主要目标是防止关键节点链路利用率过高。为了实现这一目标,提出了一种基于重路由的优化方法,通过调整部分流量的传输路径,将关键节点的负载维持在预先设定的阈值以下。

在具体实施中,首先需要明确网络的结构特征和流量需求。通信网络 $G(V,E)$ 的底层可由 IPv6 或 MPLS 网络构成,其中每条链路通过 IGP 协议或 PCE 路径计算框架确定路由路径并分发 SID 信息。链路的容量表示为 $C(E)$。流量矩阵(traffic matrix,TM)由元素 $m_{\mathrm{TM}}(u,v)$ 组成,其中 $m_{\mathrm{TM}}(u,v)$ 表示从源 u 到目的 v 的流量需求。在流量矩阵中,流量路由需将流量需求分配到源节点和目的节点之间的候选路径,以优化最大的链路利用率、网络成本或网络吞吐量等指标。在满足链路容量约束的前提下,通过在关键节点上对聚合流量进行部分重路由,使最大链路利用率值 U_{MLU} 最小化。链路利用率定义为链路负载与链路容量(带宽)的比值,而 U_{MLU} 则表示所有链路中链路利用率的最大值。

为了实现上述优化目标,本文提出了一种基于多智能体强化学习的解决方案。在本文的研究场景中,这些智能体部署在网络关键节点,根据本地状态信息对路由操作做出自主决策,从而实现流量的动态优化,有效控制关键节点链路负载压力。假设网络中节点的总数为 $|V|=n$,链路数量为 $|E|=y$。链路上的流量负载表示为 $e_y(i,j)$,$\forall i,j\in V$。L_y 表示全局网络链路负载状态信息。

3.2　SROD-LC 路由优化算法实现

在 DRL 框架中,智能体通过与环境交互实现目标学习。为了使 SROD-LC 路由优化算法适用于流量工程问题,首先需要合理设计状态空间(State)、动作空间(Action)和奖励函数(Reward)。

3.2.1　状态空间

本文将状态空间定义为从环境中获取的网络状态信息。该状态信息主要包含两部分:关键节点局部信息和全局的网络状态信息。具体而言,状态空间 s_t^k 定义如下:

$$s_t^k=(s[t],d[t],z^k[t],L_y[t]) \tag{5}$$

式中,s 表示源节点,d 表示目的节点,z^k 表示当前关键节点 k 的局部状态信息,t 为时间步。

对于全局链路负载状态信息 L_y,其由网络控制器提供,并实时推送到每个关键节点。具体而言,$L_y(i,j)$ 表示链路 (i,j) 的负载状态($\forall i,j\in V$)。对于关键节点的局部状态信息,其定义为 $z^k=[E_{\mathrm{out}}\,|\,Y^x]$,包含两部分——本地链路负载信息 E_{out} 和 SRTCS 信息 Y^x,其中,$E_{\mathrm{out}}=(k,j)$ 表示节点 k 与邻居节点 j 之间的链路负载($\forall(k,j)\in E$);Y^x 表示从节点 k 接收到的流量中,以节点 x 为当前活跃段的流量总量。

局部信息和全局信息的主要区别在于信息更新时间的差异。当使用全局链路负载时,其实时性会影响智能体决策的准确性。若信息过于陈旧,可能导致决策不一致;若更新频率过高,则可能引入额外的时间开销。关键节点可通过以下两种主要方法获取全局链路负载统计信息:①集中式方法,例如边界网关协议(border gateway protocol,BGP);②分布式遥测方法,例如分段路由带内网络遥测[33](segment routing in-band network telemetry,SR-INT)。理论上,在较大规模通信网络下,分布式遥测方法通常能够更实时地收集链路负载状态,但这超出了本文的研究范围,因此仍采用集中式方法收集链路负载状态信息。

3.2.2　动作空间

本文将动作空间定义为针对目标聚合流量的重路由操作。在 SR 网络中,每个数据包都包含一个 Segment List,其中一个 SID 是当前活跃段。数据包的转发基于其当前活跃段进行。为此,本文在关键节点 k 上部署智能体,用于执行本地重路由操作。智能体的目标是通过聚合所有具有相同活跃段的流量,生成聚合流量,并对该流量进行重路由[19]。具体而言,关键节点 k 的智能体通过识别具有相同活动段的流量,将其聚合成一个“聚合流量”,然后通过 SR 封装机制将该聚合流量重路由到备选路径。这种机制使得智能体只需针对聚合流量执行重路由操作,而无需对每个单独数据包进行复杂的流级别分类。例如,当某条链路过载时,智能体会选择重路由某个聚合流量,而非调整每个单独流。

根据状态空间信息,在关键节点 k 的智能体可以选择一个特定的动作。该动作空间定义如下:

$$a_t^k=(b[t],p[t]) \tag{6}$$

式中:b 为目标聚合流量的活跃段;p 为选择的路径,且 $p\in P_{k,x}$($P_{k,x}$ 为候选路径集合)。

当本地链路负载超过阈值 ε 时,动作触发重路由操作。触发条件如下:

$$L_y(k,j) > \varepsilon, \forall (k,j) \in E_{\text{out}} \tag{7}$$

其中,$L_y(k,j)$ 表示链路 (k,j) 的当前负载,E_{out} 表示关键节点的本地链路负载信息。

基于活跃段的聚合流量选择通过 SRTCS 统计实现。当前节点 k 所有以特定活跃段 b 为目标的流量总量被统计为聚合流量。聚合后的目标流量 b 对应于需要调整路径的流量集合。候选路径集合 $P_{k,x}$ 通过 K - 最短路径算法生成,表示计算从节点 k 到目标节点 x 的前 K 条最短路径。因此,当关键节点 k 的某条链路过载时,智能体决定将所有活跃段为 b 的流量重新路由到另一条链路。该重路由操作通过调整 Segment List 实现。

3.2.3 奖励函数

在强化学习框架中,奖励函数是引导智能体行为的关键因素,其设计直接影响算法的优化目标与性能。在多智能体强化学习场景中,多个关键节点部署 DRL 智能体以实现协作,需确保其协同工作。本文引入共享奖励机制,使多个 DRL 智能体能够协同优化其策略,从而增强对全局网络状态的适应能力。该机制不仅促进智能体间的协作,还能有效降低最大链路利用率并提升整体网络性能。

为实现有效的协作关系,设计一个基于知识共享的奖励系统,并定义由全局奖励和局部奖励两部分组成的奖励函数。全局奖励的目标是避免网络中 U_{MLU} 的增加,而局部奖励则聚焦于改善本地链路的负载状况。所有智能体的动作被视为一个联合动作,若导致 U_{MLU} 恶化,则所有智能体将受到惩罚,而非单独考虑导致 U_{MLU} 恶化的特定智能体或一组智能体。局部奖励则基于本地链路利用率在动作执行前后的比较而获得。

此外,本文还考虑了因冲突行为引发的循环对奖励机制的影响,因为循环是导致链路利用率过高的重要因素之一。若智能体的行为导致循环产生,则所有智能体将获得最大的负奖励。因此,所有智能体的动作被统称为一个共同动作,其决定了网络环境的变化。

奖励函数由全局奖励和局部奖励两部分构成,具体定义如下:

$$r_t^k = r_{\text{global}}[t] + r_{\text{local}}[t] \tag{8}$$

式中,r_{global} 表示全局奖励,r_{local} 表示局部奖励。全局奖励用于激励全局的网络性能优化,避免 U_{MLU} 的恶化;局部奖励用于优化智能体的局部链路负载。

全局奖励是基于 U_{MLU} 的变化设计,定义如下:

$$r_{\text{global}} = \begin{cases} -\alpha\Delta_{\text{MLU}}, & \Delta_{\text{MLU}} > 0 \\ 0, & \text{其他} \end{cases} \tag{9}$$

式中:α 表示全局惩罚权重,用于控制奖励的强度,取值为 300;Δ_{MLU} 表示动作执行前后 U_{MLU} 的变化量,定义为

$$\Delta_{\text{MLU}} = \max_{\forall e \in E}\left(\frac{L_y(e)_{\text{after}}}{c_e}\right) - \max_{\forall e \in E}\left(\frac{L_y(e)_{\text{before}}}{c_e}\right) \tag{10}$$

其中,c_e 表示链路 e 的链路容量,$L_y(e)_{\text{after}}$ 和 $L_y(e)_{\text{before}}$ 分别表示动作执行后和执行前通过链路 e 的流量。

局部奖励基于链路超额负载比例的变化设计,定义如下:

$$r_{\text{local}} = \begin{cases} \beta\ln(1+\gamma), & \gamma > 1 \\ -\beta\ln\left(1+\dfrac{1}{\gamma}\right), & \gamma < 1 \\ \Pi([x,P]), & \gamma = 1 \end{cases} \tag{11}$$

式中:β 为局部奖励权重,取值范围为[0,100],可根据具体场景调整;γ 表示链路利用率超额负载的改善比率,定义为

$$\gamma = \frac{\sum_{e \in E_{\text{out}}^k} \max\left(\dfrac{E_{\text{out}}^k(e)_{\text{before}}}{c_e} - \varepsilon, 0\right)}{\sum_{e \in E_{\text{out}}^k} \max\left(\dfrac{E_{\text{out}}^k(e)_{\text{after}}}{c_e} - \varepsilon, 0\right)} \tag{12}$$

其中,ε 为负载阈值,本文设置为 $\varepsilon = 60\%$。表达式 $\max\left(\dfrac{E_{\text{out}}^k(e)_{\text{before}}}{c_e} - \varepsilon, 0\right)$ 表示链路 e 在动作执行前的额外负载,若负载低于 ε 则视为 0。等式(12)右侧分式的分母表示动作执行后的总额外负载,分子表示动作执行前的总额外负载。当 $\gamma > 1$ 时,链路利用率改善,奖励值正且随 γ 递增。当 $\gamma < 1$ 时,链路利用率恶化,惩罚值负且随 γ 递减。当 $\gamma = 1$ 时,链路利用率无变化,奖励值由路径长度变化函数 $\Pi([x,P])$ 决定。该奖励函数通过区分链路负载改善和恶化的程度,有效引导智能体采取优化行为。若动作无效(无改善或恶化),奖励函数通过 $\Pi([x,P])$ 抑制不必要的重路由。

为平衡智能体在学习过程中的全局和局部奖励优化,设定参数 α 和 β 以确保最优策略学习。参数 α 需使全局奖励在 U_{MLU} 变化时提供足够惩罚,同时局部奖励引导链路负载改善。本文将 α 设为 300,增强全局奖励调节能力,优化 U_{MLU} 并避免非最优策略;β 范围设为[0,100],动态调整局

部奖励影响,确保不过强或过弱。本实验分析确定 $\beta=100$,缩放局部奖励以反映节点负载改善或恶化,避免掩盖全局奖励。这种参数设置,全局和局部奖励共同反映网络性能,为智能体提供有效指导,最终学习到降低 U_{MLU} 并改善局部负载均衡性的策略。

3.3　算法训练和部署

本文算法的训练阶段如图 1 上半部分所示。SROD-LC 训练阶段详细结构如图 2 所示。每个智能体基于本地状态定义自己的动作,所有智能体动作的组合构成联合动作。前文已详细描述了智能体的实现。本研究采用 PPO 算法模型来构建 SROD-LC 路由优化算法。通过训练并部署智能体,实现关键节点负载的动态路径优化调整。

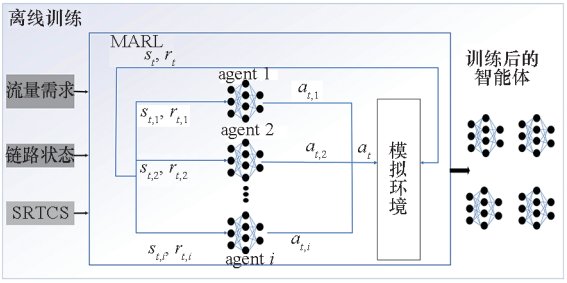


图 2　SROD-LC 训练阶段详细结构

Fig. 2　Detailed structure of the SROD-LC training phase

PPO 是一种基于策略梯度的强化学习算法,通过限制策略更新的幅度,实现算法稳定性和收敛速度的有效平衡。其核心思想在于优化智能体的策略函数 $\pi_\theta(a|s)$,以最大化累积奖励 r_t,同时通过引入裁剪函数来约束策略更新的幅度。PPO 的优化目标函数可表示为:

$$L^{\mathrm{ppo}}(\theta)=E_t\{\min[r_t(\theta)A_t,clip(\varphi_t(\theta),1-\psi,1+\psi)A_t]\} \tag{13}$$

式中:$\varphi_t(\theta)=\dfrac{\pi_\theta(a_t|s_t)}{\pi_{\theta\mathrm{old}}(a_t|s_t)}$ 表示动作的概率比值;A_t 是优势函数,用于衡量动作在给定状态下的优劣性;ψ 是裁剪范围,用于限制策略的更新幅度;$clip(\cdot)$ 为用于限制策略更新幅度的裁剪函数。在 SROD-LC 路由优化问题中,智能体以本地状态(包括链路负载、SRTCS 信息以及控制器推送的全局链路利用率信息)作为输入,通过策略网络生成路径调整方案($\pi_\theta(s_t)=a_t$)。智能体与环境进行迭代训练,直到模型收敛。训练完成后,模型被部署在实际网络中的关键节点,形成分布式协作的路由优化方案。算法 1 详细描述了 SROD-LC 路由优化算法的实现过程。

算法 1　SROD-LC 路由优化算法

Alg. 1　SROD-LC routing optimization algorithm

输入:状态空间 State,奖励函数 Reward,$m_{\mathrm{TM}}(u,v)$

输出:训练完成的 DRL 参数

1. 初始化 Critic 网络 Q_{C} 和 Actor 网络 Q_{a} 以及 $\pi_\theta(s)$;
2. 初始化环境 env 以及经验回放池 R;
3. **if**　PPO 收敛　/* PPO 决策 */
4.　**if**　本地链路负载超过阈值 ε;
5.　　在环境执行动作 a_t,获得奖励 r_t;
6.　　结果$\{s_t,a_t,r_t,s_{t+1}\}$存放到重放缓存 R;
7.　**end if**
8. **else**　/* 线下训练 */
9.　**for** 迭代回合数(episode)
10.　　初始化 R,重置 env;
11.　　根据当前 Actor 网络的策略 $\pi_\theta(s)$执行动作 a_t(在 env);
12.　　运行 K 个轮次,收集$\{s_t,a_t,r_t,s_{t+1}\}$并存放到 R;
13.　　基于当前的值函数,计算获得优势函数 A_t;
14.　　**for** 1 到子迭代回合 M
15.　　从 R 中采样最小批次个数据$\{s_t,a_t,r_t,s_{t+1}\}$;
16.　　计算 PPO 目标 $L'(Q_{\mathrm{a}})$;
17.　　基于式(13),更新 Actor 网络参数 Q_{a};
18.　　通过 Critic 网络获取预测值 $V(s)$;
19.　　通过均方误差(MSE)计算价值损失来拟合价值函数 $L'(Q_{\mathrm{C}})$;
20.　　更新 Critic 网络参数 Q_{C};
21.　**end for**
22.　**end for**
23. **end if**

SROD-LC 算法部署流程如下:离线训练完成后,将 PPO 模型的 Actor 和 Critic 网络参数导出为轻量级格式,并通过剪枝技术压缩模型以适配路由器资源。接着,利用 SDN 控制器收集网络拓扑和流量数据,基于负载中心性理论选取关键节点,设置变化阈值 $\mu=50\%$ 以控制更新频率。随后,通过安全文件传输协议或 SDN 管理通道将模型分发至支持 SR 的路由器,运行轻量级推理引擎,加载 PPO 模型,初始化状态空间(本地链路负载、SRTCS 信息、全局链路状态),并配置 SR 控制平面以支持动态 Segment List 调整。最后,通过测试流量矩阵验证智能体能否基于本地状态和控制器推送的全局信息生成重路由动作,确保链路负载超 60% 时触发重路由,从而降低 U_{MLU},优化网络性能。

4 实验与结果分析

4.1 实验环境

实验是在一台配备 Intel i7 – 13790F CPU(16 核)、32 GB RAM 并运行 Ubuntu 16.04 操作系统的服务器上进行。本文算法实现于 Python3.0 和 PyTorch 框架之上,模拟实验环境构建于 OMNeT ++平台,并借助 Inet 库进行功能扩展。研究中对 SROD-LC 算法的网络性能进行了系统性评估。实验采用真实的网络拓扑 GEANT[34] 和 germany50[35],其拓扑信息数据来自收集和共享互联网拓扑数据的 Internet Topology Zoo 项目平台。具体而言,GEANT 拓扑共有 34 个节点和 52 条链路,而 germany50 拓扑则由 50 个节点和 176 条链路组成。在性能评估中,流量矩阵数据集采用重力模型生成,实验共构建了 1 000 个流量矩阵,其中 700 个用于训练集、300 个用于测试集。容量规划基于过度供应策略执行,以确保初始链路利用率低于预设阈值。

4.2 基准算法

针对网络拥塞或链路故障的动态响应问题,现有研究方案主要分为两类:基于控制器的方案和基于本地设备的方案。值得注意的是,近期研究多聚焦于本地设备的方案。为了全面评估本文提出的 SROD-LC 路由优化算法的网络性能,本文对以下算法进行了对比分析:

1)Original[14]:该算法通过控制器计算备选 SR 策略缓解网络拥塞,可实现流量自动响应和适应性调整,但设备与控制器交互导致重新配置时延。集中式方案有效,但可扩展性和响应速度受限。SROD-LC 采用分布式 MARL 框架和基于负载中心性的节点选择,提供更快、更灵活的优化。

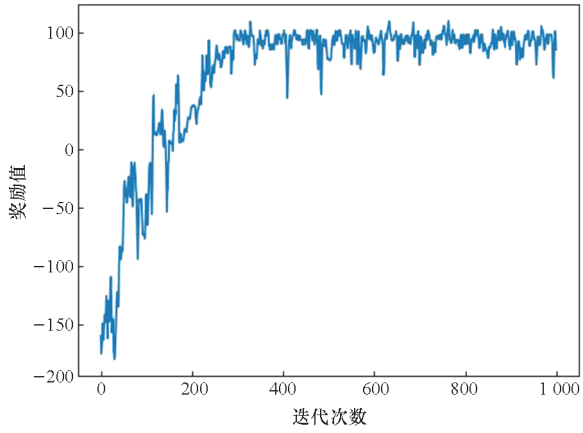
2)SR-LILLC[25]:该算法采用 MARL 的本地智能体方法,利用 DQN 实现 SR 网络链路负载自主控制,通过实时监测链路负载状态触发重路由优化性能。但该方案未集成 LC 以动态感知负载。SROD-LC 结合 LC 与 PPO 的 MARL,显著提升 U_{MLU} 优化效果。

3)CM[17]:该算法提出了一种分布式拥塞管理机制,利用 SR 通过 UCMP 技术将拥塞链路的流量部分重路由至替代路径。然而,该方法依赖全局信息,且 UCMP 在 SR 多播中的计算开销较大。相比之下,SROD-LC 通过将分布式代理部署于负载中心性识别的关键节点,显著降低了计算开销并提升了优化效率。

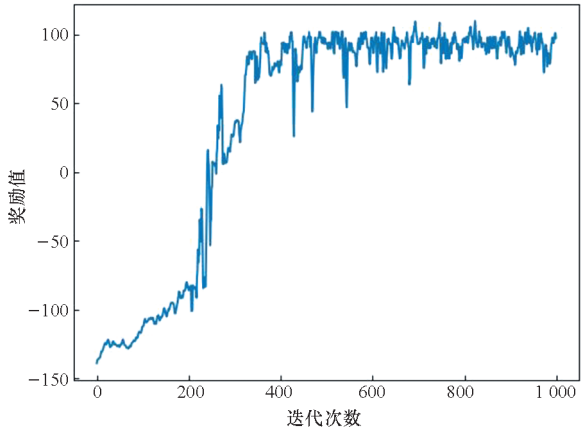
4.3 实验结果分析

4.3.1 算法的收敛性分析

实验验证了 SROD-LC 方法的收敛性,具体分析了训练奖励值随训练集增长的变化趋势。图 3 展示了相应的奖励值曲线。从图 3 可以看出,训练奖励值曲线随着迭代次数的增加呈现波动变化。在图 3(a)所示的 GEANT 通信场景中,SROD-LC 的奖励值曲线在迭代次数小于 380 时波动显著,而当迭代次数超过 380 时,奖励值曲线趋于平缓并维持在一个稳定区间内。在图 3(b)所示的 germany50 通信场景中,SROD-LC 方法在迭代次数达到 400 时开始收敛,当迭代次数超过 400 时,奖励值曲线同样趋于平缓。



(a) GEANT



(b) germany50

图 3 奖励值随训练迭代次数的变化趋势

Fig. 3 Reward value variation trend as iteration times increase

从图 3(a)和图 3(b)所示的两个通信场景的收敛结果可见,本文提出的 SROD-LC 路由优化算法在训练过程中具备良好的收敛性。

4.3.2 关键节点选取分析

不同关键节点比例及其选择方案对网络性能的影响是值得探索的问题。鉴于理论分析难以确

定最优关键节点比例和关键节点方案,本文将通过实验测试评估不同关键节点比例及选择方案的网络性能。为此,本文设计了两个典型通信网络场景,通过实验测试和性能评估,比较了不同选择方案在典型场景下的网络性能表现,为关键节点的选择方案和比例提供了重要参考。基于此,本文考虑了以下关键节点选择方案:

1)SROD-BC:基于介数中心性的关键节点选择方法,指在通信网络拓扑中,根据节点的介数中心性(betweenness centrality, BC)值选择关键节点。

2)SROD-DC:基于度中心性的关键节点选择方法,指在通信网络拓扑中,根据节点的度中心性(degree centrality,DC)值选择关键节点。

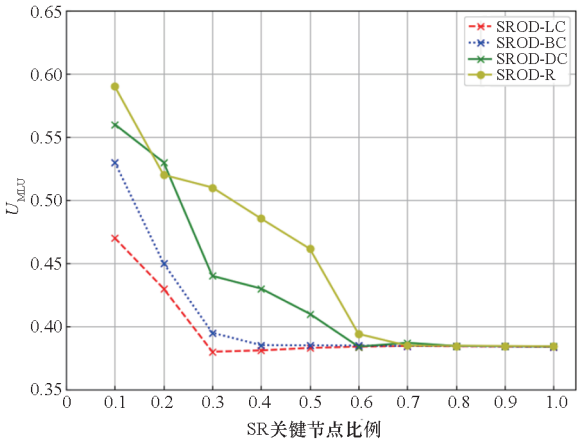
3)SROD-R:随机关键节点选择方法,指在通信网络拓扑中,采用随机(random)方式选择关键节点。

4)SROD-LC:基于链路负载中心性的关键节点选择方法,指在通信网络拓扑中,根据节点的链路负载中心性进行关键节点的选择。

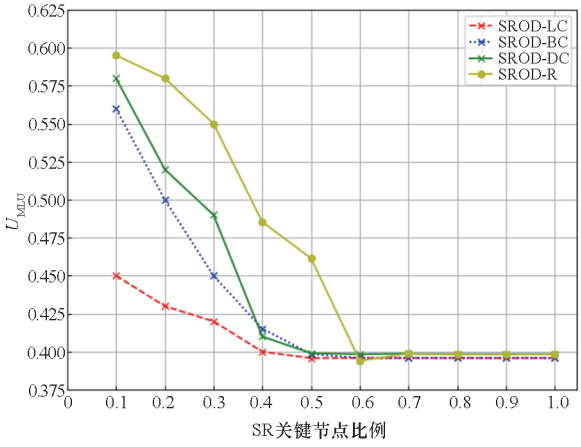
上述 SROD-BC/DC/LC 方法通过不同中心性指标量化节点在网络中的重要性,从而实现关键节点的选择。其中,SROD-BC 和 SROD-DC 侧重于网络的结构特征,而 SROD-LC 则关注网络的动态特征。通过分析关键节点选择比例对网络性能的影响,本文旨在确定实现良好网络性能所需的合理选择比例。

实验结果如图 4 所示,SROD-LC 的关键节点选择方案在提升网络性能方面具有显著优势。该方案通过优先选择链路负载较高的关键节点,在网络流量较大的情况下能够更有效地进行负载调整,从而避免链路过载或瓶颈现象的发生。与仅基于节点中心性(SROD-BC 和 SROD-DC)的方法相比,SROD-LC 不仅考虑了节点的重要性,还充分考虑了链路的负载情况,因此在流量高峰时能够实现更有效的流量重路由调整。而 SROD-R 采用随机选择关键节点的方式,其性能表现最差。SROD-LC 通过关注关键节点链路负载,在网络负载增加时通过重路由部分流量有效降低了关键链路的负载,从而避免了过载现象。由此可见,SROD-LC 方法在性能上优于其他方法。

此外,图 4(a)和图 4(b)进一步展示了不同关键节点选择方案在两种通信网络场景下的 U_{MLU} 与关键节点比例的关系。在两种场景中,随着关键节点比例的增加,SROD-LC 的 U_{MLU} 均保持在较低的水平,相较于其他方法表现出更好的链路利用率控制能力。然而,当 SR 关键节点比例进一步增加时,SROD-LC 的改进效果逐渐减弱。从图 4(a)和图 4(b)可以看出,SR 关键节点比例在 0.3 ~ 0.5 之间时能够达到最佳改进效果。因此,SROD-LC 基于链路负载中心性的节点选择方案在不同网络场景中都表现出了优越的性能,不仅能够有效控制 U_{MLU},而且在关键节点选择比例保持在 0.3 ~ 0.5 时能够实现较好的网络性能。



(a) GEANT



(b) germany50

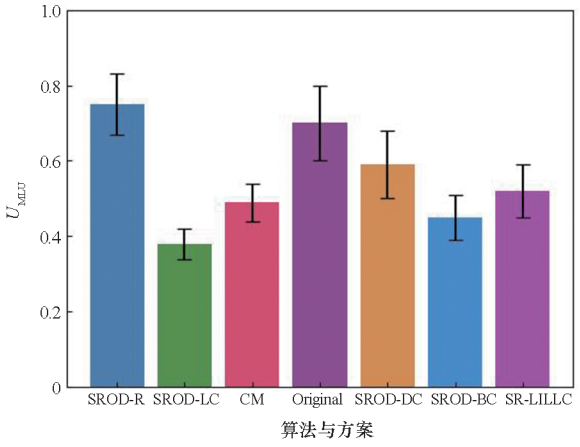
图 4　不同关键节点比例与选取方案的性能对比

Fig. 4　Performance comparison of different key node selection ratios and schemes

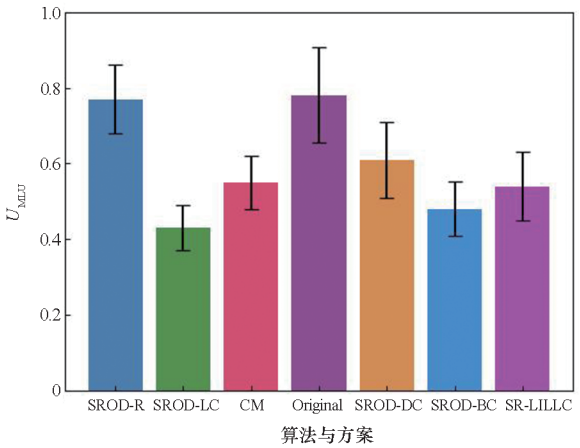
4.3.3　算法效果分析

本小节将评估所提出的 SROD-LC 路由优化算法的网络性能,并与上述基准路由算法进行比较,其中关键节点选择比例设置为 0.5。实验对两种网络拓扑场景下各种算法及不同关键节点选择方案的网络进行了 5 次状态采样,并计算其 U_{MLU} 性能。图 5 展示了不同算法及方案的网络 U_{MLU} 性能,其中条形图表示平均性能,误差线则显示了测试结果的最大值和最小值范围。

图 5(a)和图 5(b)分别展示了在 GEANT 和 germany50 两种网络场景下,不同 SR 路由优化算



（a）GEANT



（b）germany50

图 5　各种 SR 路由算法的性能对比

Fig. 5　Performance comparison of various SR routing algorithms

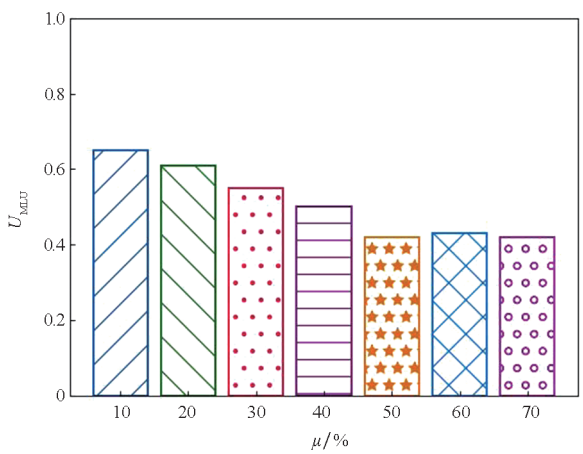
法的 U_{MLU} 性能比较。在两个场景中，SROD-LC 均表现出较低的 U_{MLU} 值，表明其在减少关键节点链路过载方面具有良好的性能。具体而言，在 GEANT 场景下，SROD-LC 的 U_{MLU} 性能分别比 CM 和 SR-LILLC 提升了 22% 和 35%；在 germany50 场景下，其性能提升幅度分别为 21% 和 32%。与其他算法相比，SROD-LC 算法在 U_{MLU} 性能上表现更优。

SROD-LC 算法通过优先选择链路负载较高的关键节点，能够有效平衡网络流量，避免特定链路过载。相比之下，SROD-DC 和 SROD-BC 算法在选择关键节点时更侧重于网络拓扑属性。SR-LILLC 使用 DQN 强化学习方法进行多智能体控制，但在大部分场景下性能表现不如 SROD-LC 算法，因为其智能体配置基于拓扑度指标，未考虑链路负载情况。Original 算法依赖于网络控制器和路由设备信息交互，重新配置操作相对较慢，导致其 U_{MLU} 值较高，未能及时响应流量变化。CM 算法通过 UCMP 重路由部分流量，但无法避免重新选择到负载较高的链路，导致部分链路上的流量依然过载。因此，SROD-LC 算法提供了一种有效的路由优化思路，能够在上述两个通信网络场景下有效减少关键链路过载，从而提升网络的整体性能。

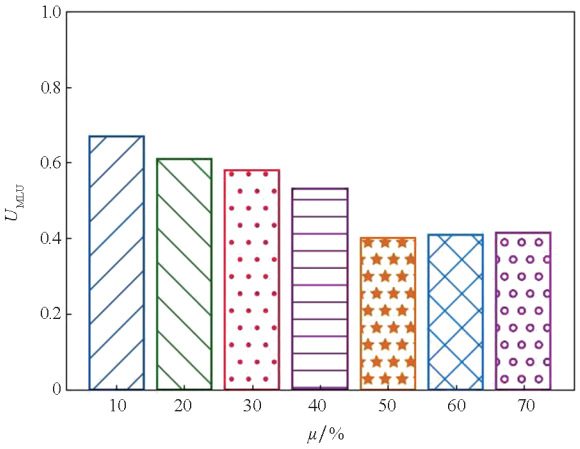
4.3.4　关键节点变化阈值 μ 的参数敏感性分析

在 SROD-LC 算法中，关键节点的选取基于负载中心性理论，通过计算每个节点的 C_{LC} 值来量化其在网络流量分布中的重要性。C_{LC} 值较高的节点被认为是关键节点，因为它们对网络的负载平衡和拥塞管理具有显著影响。链路负载中心性的动态变化会导致关键节点的选择发生调整，而关键节点变动阈值 μ 的选择可能引发频繁的线下训练调整，进而影响优化效果。本研究引入了一个阈值 μ，仅当关键节点的变化量（基于初始选择）超过该阈值时，才执行增量更新以及线下训练。但鉴于理论分析难以确定关键节点变动阈值，本文在两个典型通信网络场景中进行了实验研究，比较了不同关键节点变动阈值 μ 的网络性能，为 SROD-LC 方法的关键节点变动阈值选择提供有价值的参考。

图 6（a）和图 6（b）分别展示了在 GEANT 和 germany50 两种网络场景下，不同阈值 μ 的 U_{MLU} 性能比较。实验选取 μ 的范围为 10% ~ 70%。在两个场景中，μ = 50% 表现出较低的最大链路利用率，说明其在缓解关键节点链路过载方面具有显著优势。具体而言，当变化率 μ 在 50% ~ 70% 之间时，SROD-LC 的网络性能保持相对稳定；相反，当 μ 低于 50% 时，SROD-LC 的网络性能较差。对于比例为 70% 之后的情况，其性能与 50% 时的性能相差不大，因此在图中未示出。网络性能结果与研究的预期一致，表明 μ 值在 SROD-LC 中起着关键作用。如果 μ 值太低，关键



（a）GEANT



(b) germany50

图 6　不同关键节点变动阈值 μ 的网络性能对比

Fig. 6　Comparison of network performance with different key node change thresholds μ

节点变动就会变得频繁,限制了 SROD-LC 管理整个网络的能力,从而影响网络性能。当 μ 值进一步增加时,SROD-LC 的改进效果逐渐减弱,没有实际意义。从图 6 中可以进一步观察到,当 μ 值为 50% 时,网络性能保持相对较好,表明在此范围内的 μ 值足以有效地优化网络性能,因此,在 SROD-LC 中,实验最终选择 μ 值为 50%。

4.3.5　算法时间开销分析

为了评估 SROD-LC 路由算法的路由决策时间开销,本文测量了不同基准算法 SR 路由决策的执行时间,即新路由路径规则生成所需的时间。为确保测量结果的有效性,实验在相同的仿真环境中测量了算法时间开销,其中关键节点比例设置为 0.5。通过变化输入流量矩阵,实验对每种算法对应智能体进行了 5 次测试,记录了决策过程时间开销的最大值和最小值。表 1 列出了各种 SR 路由算法的时间开销范围。

表 1　各种 SR 路由算法的时间开销范围

Tab. 1　Time cost range of various SR routing algorithms

单位:ms

算法	GEANT			germany50		
	t_{min}	t_{max}	$t_{Average}$	t_{min}	t_{max}	$t_{Average}$
Original	125.2	135.5	130.8	138.5	155.9	148.2
SR-LILLC	26.4	36.6	31.7	32.1	46.6	38.3
CM	19.6	26.4	22.8	26.5	38.8	33.8
SROD-LC	22.5	32.2	23.7	29.2	42.6	30.5

从表 1 可以看出,不同 SR 路由算法在网络路由的时间开销上存在显著差异。SROD-LC 算法的时间开销与 SR-LILLC 算法较为相近,但相较于 Original 算法,其时间开销显著降低。然而,与 CM 算法相比,SROD-LC 算法时间的最大与最小开销仍然较高。以下将从多个角度对上述结果进行分析。

Original 算法依赖集中式网络控制器和全局计算,且每次重路由需要控制器的参与,导致时间开销较高。CM 算法通过 UCMP 计算多个备选路径并进行流量重路由,其时间开销低于 SR-LILLC 和 SROD-LC 算法。SR-LILLC 算法采用了本地智能体 DQN 进行自主决策,减少了对全局控制的依赖,且在训练完成后,其执行效率较高,时间开销较小。SROD-LC 则采用本地智能体 PPO 算法进行自主路由决策,因此在时间开销上与 SR-LILLC 相差不大。相比在更大的 germany50 拓扑,就平均而言,SROD-LC 相比 CM 更具优势。

5　结论

本文提出的 SROD-LC 算法缓解了大规模通信网络中关键节点链路利用率过高、快速响应能力不足导致网络拥塞和性能下降的问题。SROD-LC 通过负载中心性理论动态量化关键节点重要性,并利用分布式 DRL 智能体实现自主路由决策,显著提升响应速度。仿真实验结果表明,SROD-LC 在最大链路利用率指标上优于 CM、SR-LILLC 和 Original 等基准算法。此外,通过离线训练与在线部署相结合,SROD-LC 算法能够将路由决策时间控制在毫秒级,有效平衡了动态网络的实时性与稳定性。

目前,基于 MARL 的路由优化已成为研究热点,但分布式 MARL 中智能体实时获取全局信息以实现协作仍面临挑战。依赖网络控制器获取全局信息会增加其负担,且信息传输至智能体也将带来额外时间开销。为此,我们将引入带内网络遥测(in-band network telemetry, INT)技术,以实现全局信息的快速获取,以增强分布式 MARL 在路由优化中的应用效果。

参考文献(References)

[1] WANG N, HO K, PAVLOU G, et al. An overview of routing optimization for Internet traffic engineering [J]. IEEE Communications Surveys & Tutorials, 2008, 10(1): 36 - 56.

[2] ABDULLAH Z N, AHMAD I, HUSSAIN I. Segment routing in software defined networks: a survey [J]. IEEE Communications Surveys & Tutorials, 2019, 21(1): 464 -

486.

[3] BOLDRINI L, BACHIDDU M, KONING R, et al. User controlled routing exploiting PCEPS and inter-domain label switched paths[C]//Proceedings of the Second International Conference on Advances in Computing Research (ACR'24), 2024: 465 – 478.

[4] FOERSTER K T, PARHAM M, CHIESA M, et al. TI-MFA: keep calm and reroute segments fast[C]//Proceedings of the IEEE INFOCOM 2018: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2018: 415 – 420.

[5] GIORGETTI A, SGAMBELLURI A, PAOLUCCI F, et al. Demonstration of dynamic restoration in segment routing multi-layer SDN networks[C]//Proceedings of the Optical Fiber Communication Conference, 2016: Th4G.4.

[6] GIORGETTI A, SGAMBELLURI A, PAOLUCCI F, et al. Segment routing for effective recovery and multi-domain traffic engineering[J]. Journal of Optical Communications and Networking, 2017, 9(2): A223.

[7] Cisco Systems, Inc. Cisco crosswork optimization engine 4.1 user guide: user local congestion mitigation (LCM) to mitigate network congestion locally[EB/OL]. (2022 – 10 – 31)[2025 – 05 – 10]. https://www.cisco.com/c/en/us/td/docs/cloud-systems-management/crosswork-optimization-engine/4 – 1/UserGuide/b _ cisco-crosswork-optimization-engine-4_1_userguide/m_mitigate-congestion-locally.html.

[8] HUAWEI Technologies Co. HUAWEI iMaster NCE-IP V100R023C10 data sheet (enterprise)[EB/OL]. (2024 – 04 – 08)[2025 – 05 – 10]. https://e.huawei.com/en/material/enterprise/2d6abf7b94d94c3c862725e6997dcb99.

[9] VENTRE P L, SALSANO S, POLVERINI M, et al. Segment routing: a comprehensive survey of research activities, standardization efforts, and implementation results[J]. IEEE Communications Surveys & Tutorials, 2021, 23(1): 182 – 221.

[10] GOH K I, KAHNG B, KIM D. Universal behavior of load distribution in scale-free networks[J]. Physical Review Letters, 2001, 87(27): 278701.

[11] WU D, CUI L. A comprehensive survey on segment routing traffic engineering[J]. Digital Communications and Networks, 2023, 9(4): 990 – 1008.

[12] ROELENS L, DE DIOS Ó G, DE MIGUEL I, et al. Performance evaluation of TI-LFA in traffic-engineered segment routing-based networks[C]//Proceedings of the 19th International Conference on the Design of Reliable Communication Networks (DRCN), 2023: 1 – 8.

[13] LEMESHKO O, YEREMENKO O. Linear optimization model of MPLS Traffic Engineering Fast ReRoute for link, node, and bandwidth protection[C]//Proceedings of the 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), 2018: 1009 – 1013.

[14] SHEU J P, CHEN Y C. A scalable and bandwidth-efficient multicast algorithm based on segment routing in software-defined networking[C]//Proceedings of the IEEE International Conference on Communications (ICC), 2017: 1 – 6.

[15] BRUNDIERS A, SCHÜLLER T, ASCHENBRUCK N. Midpoint optimization for segment routing[C]//Proceedings of the IEEE INFOCOM 2022: IEEE Conference on Computer Communications, 2022: 1579 – 1588.

[16] BRUNDIERS A, SCHÜLLER T, ASCHENBRUCK N. Tactical traffic engineering with segment routing midpoint optimization[C]//Proceedings of the IFIP Networking Conference (IFIP Networking), 2023: 1 – 9.

[17] MEDAGLIANI P, MARTIN S, MAGNOUCHE Y, et al. Distributed tactical TE with segment routing[J]. IEEE Transactions on Network and Service Management, 2024, 21(5): 4974 – 4987.

[18] 高新成, 刘威, 王启龙, 等. 基于SDN的混合分段路由概率流调度机制[J]. 计算机应用研究, 2023, 40(11): 3382 – 3387.
GAO X C, LIU W, WANG Q L, et al. SDN-based hybrid segmented routing probabilistic flow scheduling mechanism[J]. Application Research of Computers, 2023, 40(11): 3382 – 3387. (in Chinese)

[19] WANG H N, LIU N, ZHANG Y Y, et al. Deep reinforcement learning: a survey[J]. Frontiers of Information Technology & Electronic Engineering, 2020, 21(12): 1726 – 1744.

[20] TIAN Y, WANG Z L, YIN X, et al. Traffic engineering in partially deployed segment routing over IPv6 network with deep reinforcement learning[J]. ACM Transactions on Networking, 2020, 28(4): 1573 – 1586.

[21] CHEN B, SUN P H, ZHANG P, et al. Traffic engineering based on deep reinforcement learning in hybrid IP/SR network[J]. China Communications, 2021, 18(10): 204 – 213.

[22] REN B B, GUO D K, YUAN Y L, et al. Optimal deployment of SRv6 to enable network interconnection service[J]. ACM Transactions on Networking, 2022, 30(1): 120 – 133.

[23] TONG V, SOUIHI S, TRAN H A, et al. SDN-based application-aware segment routing for large-scale network[J]. IEEE Systems Journal, 2022, 16(3): 4401 – 4410.

[24] AURELI D, CIANFRANI A, LISTANTI M, et al. Intelligent link load control in a segment routing network via deep reinforcement learning[C]//Proceedings of the 25th Conference on Innovation in Clouds, Internet and Networks (ICIN), 2022: 32 – 39.

[25] CIANFRANI A, AURELI D, LISTANTI M, et al. Multi agent reinforcement learning based local routing strategy to reduce end-to-end delays in segment routing networks[C]//Proceedings of the IEEE INFOCOM 2023: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2023: 1 – 6.

[26] WANG L H, LU L, WANG M, et al. SNS: smart node selection for scalable traffic engineering in segment routing networks[J]. IEEE Transactions on Network and Service Management, 2025, 22(1): 92 – 106.

[27] 赵鹏程, 于俊清, 李冬. 一种基于深度学习的SRv6网络

流量调度优化算法[J]. 信息网络安全, 2024, 24(2): 272 – 281.
ZHAO P C, YU J Q, LI D. An optimal algorithm for traffic scheduling in SRv6 network based on deep learning[J]. Netinfo Security, 2024, 24(2): 272 – 281. (in Chinese)

[28] VENTRE P L, TAJIKI M M, SALSANO S, et al. SDN architecture and southbound APIs for IPv6 segment routing enabled wide area networks[J]. IEEE Transactions on Network and Service Management, 2018, 15(4): 1378 – 1392.

[29] LI Z B, HU Z B, LI C. SRv6 network programming: ushering in a new era of IP networks[M]. Florida: CRC Press, 2021.

[30] LIU N, JIA C B, HOU B N, et al. 6Search: a reinforcement learning-based traceroute approach for efficient IPv6 topology discovery[J]. Computer Networks, 2023, 235: 109987.

[31] POLVERINI M, CIANFRANI A, LISTANTI M. A theoretical framework for network monitoring exploiting segment routing counters[J]. IEEE Transactions on Network and Service Management, 2020, 17(3): 1924 – 1940.

[32] MACCARI L, GHIRO L, GUERRIERI A, et al. Exact distributed load centrality computation: algorithms, convergence, and applications to distance vector routing[J]. IEEE Transactions on Parallel and Distributed Systems, 2020, 31(7): 1693 – 1706.

[33] ZHENG Q T, TANG S F, CHEN B F, et al. Highly-efficient and adaptive network monitoring: when INT meets segment routing[J]. IEEE Transactions on Network and Service Management, 2021, 18(3): 2587 – 2597.

[34] KNIGHT S, NGUYEN H X, FALKNER N, et al. The Internet topology zoo[J]. IEEE Journal on Selected Areas in Communications, 2011, 29(9): 1765 – 1775.

[35] ORLOWSKI S, WESSÄLY R, PIÓRO M, et al. SNDlib 1.0: survivable network design library[J]. Networks, 2010, 55(3): 276 – 286.