融合SDN与目标导向分层强化学习的覆盖组播路由方法

叶苗[1,2]，李繁有[1]，文鹏[1]，蒋秋香[1]，王勇[2]，何倩[2]，叶聪[1]
（1.桂林电子科技大学信息与通信学院，广西 桂林 541004；
2.桂林电子科技大学云网融合与数据安全广西高校工程研究中心，广西 桂林 541004）

摘　要：针对传统网络架构下覆盖组播缺乏对底层网络状态的感知、难以适应网络动态变化，以及现有强化学习方法因覆盖组播树路径耦合、面临问题规模大、动作空间维度高导致学习不稳定、收敛缓慢等问题，提出一种基于目标导向分层强化学习的智能覆盖组播路由方法GOHRL-OM。首先，利用SDN的全局感知能力，构建动态流量矩阵为路由决策提供全局信息支撑。其次，GOHRL-OM结合目标导向强化学习与分层强化学习优化覆盖组播树：目标导向机制引入任务目标，增强策略学习方向性；分层学习将任务分解为上下层子任务，通过协同策略和分层奖励实现任务解耦与分层优化，从而降低动作维度并提升学习稳定性。仿真实验表明，相较于现有方法，GOHRL-OM在优化吞吐量、时延与丢包率的同时，具备更加灵活的路由决策和网络适应能力。

关键词：软件定义网络；目标导向；分层强化学习；覆盖组播
中图分类号：TP393.0
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2026002

Goal-oriented hierarchical reinforcement learning-based overlay multicast routing method with SDN integration

Ye Miao[1,2], Li Fanyou[1], Wen Peng[1], Jiang Qiuxiang[1], Wang Yong[2], He Qian[2], Ye Cong[1]
1. School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China
2. Guangxi University Engineering Research Center of Cloud Network Convergence and Data Security, Guilin University of Electronic Technology, Guilin 541004, China

Abstract: Traditional overlay multicast lacks effective awareness of underlying network states and adaptability to dynamic network conditions. Meanwhile, existing reinforcement learning–based routing methods often suffer from unstable learning and slow convergence due to coupled multicast tree paths and high-dimensional action spaces. To address these issues, this paper proposes an intelligent overlay multicast routing method based on goal-oriented hierarchical reinforcement learning, termed GOHRL-OM. By exploiting the global network visibility provided by software-defined networking (SDN), a dynamic traffic matrix is constructed to support routing decisions with global information. GOHRL-OM integrates goal-oriented learning with hierarchical task decomposition to decouple routing optimization, reduce action dimensionality, and enhance learning stability. Simulation results show that, compared with existing methods, GOHRL-OM achieves improved throughput, lower delay, and reduced packet loss, while adapting effectively to dynamic network environments.

Keywords: software defined network, goal-oriented, hierarchical reinforcement learning, overlay multicast

---

收稿日期：2025–09–10；**修回日期：**2025–12–12
通信作者：何倩，heqian@guet.edu.cn
基金项目：国家自然科学基金资助项目（No.62161006, No.62372353）；广西研究生教育创新计划基金资助项目（No.YCSW2025351）；桂林电子科技大学云网融合与数据安全广西高校工程研究中心（No.C25KYS00RX01）；认知无线电与信息处理教育部重点实验室主任基金资助项目（No.CRKL220103）
Foundation Items: The National Natural Science Foundation of China (No.62161006, No.62372353), The Subsidization of Innovation Project of Guangxi Graduate Education (No.YCSW2025351), Guangxi University Engineering Research Center of Cloud Network Convergence and Data Security, Guilin University of Electronic Technology (No.C25KYS00RX01), Key Laboratory of Cognitive Radio and Information Processing, Ministry of Education (No.CRKL220103)

0　引言

近年来，随着互联网应用的快速发展和数据流量的持续增长，高效实现大规模数据分发已成为多播通信研究[1]的热点问题。其中，IP组播通过路由器复制与减少转发数据[2]，能够有效降低网络带宽资源开销与传输时延。然而，其实际应用存在诸多限制：首先，它依赖路由器为每个多播维护状态信息，违背IP层“无状态”设计原则，增加网络设备负担与协议复杂度；其次，在可靠性、拥塞控制和安全机制的实现上相较于单播更为复杂；此外，由于不同互联网服务提供商（Internet service provider, ISP）之间缺乏统一部署，IP组播往往局限于局部网络，难以实现跨域应用。网络状态频繁变化、链路不稳定等因素进一步加剧了组播维护难度，限制了其在现代互联网中的推广与应用[3]。

覆盖组播（overlay multicast, OM）作为一种更具灵活性的替代方案，通过在应用层工作的终端节点构建逻辑组播结构（如树状、网状或混合覆盖拓扑[4]），利用底层单播实现数据传输，不需要改动网络基础设施即可实现类似组播的高效数据转发。在OM架构中，终端节点既是数据接收者，也可作为中继节点参与数据转发，增强了多点通信结构的灵活构建与路由的动态调整。OM凭借对异构跨域网络的兼容性以及易部署等优势，有效弥补了IP组播在部署方面的不足，被广泛应用于P2P流媒体[5]、在线会议平台[6]、边缘计算[7]、联邦学习[8]等场景。

尽管OM克服了IP组播在部署与兼容性方面的部分限制[9-10]，但在网络流量高度动态变化场景下，构建高效、稳定且可扩展的OM结构[11]仍面临挑战：首先，最优覆盖组播树构建本质上属于NP难的组合优化问题[12-13]；其次，现有OM技术普遍依赖终端节点，缺乏对底层网络状态的实时感知，难以实现全局最优路径选择与资源调度。随着网络状态的快速变化，原有的覆盖组播结构往往难以适应链路负载波动与拓扑结构变更，易导致链路拥塞与传输性能下降。因此，最优覆盖组播树构建应融合网络状态感知与高效求解算法，才能实现对网络资源的精细管理和高效利用。

传统网络架构采用分布式决策模式以及受边界网关协议等机制的限制，难以获取全局网络状态。使OM在传统网络架构下难以直接获取底层网络状态[14]，缺乏全局视角与集中控制能力，易导致网络负载不均衡等问题。相比之下，软件定义网络（software defined network, SDN）通过解耦控制平面与数据平面[15]，实现集中控制与全局可视。将SDN与覆盖网络相结合，SDN可弥补基于终端节点的覆盖网络在网络状态感知方面的不足，而覆盖网络可增强SDN部署域在数据平面互联方面的能力[16]。近期研究Li等[17]提出了基于位索引显式复制流量工程的无状态覆盖组播方法，但该方法仅关注时延或带宽等单一性能指标，缺乏对多服务质量（quality of service, QoS）性能指标的综合优化，且基于局部最优的最小生成树算法亦难以适应高速动态变化的网络环境。

而针对NP难的最优覆盖组播树组合优化问题，已有研究提出了近似优化、启发式搜索、智能重构等方法。Zhu等[12]提出了基于瓶颈边替换的中心化算法和基于gossip协议的分布式重构机制，可缓解带宽瓶颈但重构代价高。Banik等[13]设计了链式时延匹配算法，在多项式时间内构建最小化时延差异的近似覆盖组播结构，但适应性差且存在冗余。Banerjee等[18]提出了覆盖组播网络架构（overlay multicast network infrastructure, OMNI）结合局部变换与模拟退火，但易陷入局部最优。总体而言，这些方法普遍存在计算复杂度高、收敛速度慢、对动态网络适应性不足等问题，难以满足高效稳定的覆盖组播需求。

相比以上求解最优覆盖组播树的传统近似优化与启发式搜索方法，强化学习（reinforcement learning, RL）作为数据驱动技术，通过与环境交互从网络中学习覆盖组播路由决策策略[19]，在提升求解效率的同时也具备良好的动态自适应能力，适用于高度变化的网络环境。尤其在SDN架构下，控制器可实时获取全局网络状态，为RL方法应用在求解最优覆盖组播树问题上提供支撑。目前，已有研究将RL方法应用于单播及IP组播路由问题[20-22]。然而，由于覆盖组播工作于应用层，其路径构建依赖于端节点间的逻辑连接，直接采用RL方法仍面临奖励稀疏、求解问题规模维度高和动作空间维度大等挑战，使训练收敛缓慢，难以满足路由实时调整的需求。

近年来，在深度强化学习领域涌现出目标导向

强化学习、分层强化学习等新方法。这类方法可通过任务分解或目标引导缓解复杂路由问题中的奖励稀疏与高维决策困难[23-25]等问题。然而，现有方法主要面向单播或IP组播的场景，在覆盖组播中仍面临以下局限：目标导向强化学习方法依赖合理的子目标设定，但覆盖组播路径高度耦合，目标选择困难，易导致子目标不可达、学习过程不稳定和样本利用率降低；分层强化学习方法虽可通过分解决策过程缓解路径耦合、降低问题规模的复杂度，但上下层奖励设计与策略协调困难，易陷入局部最优。因此，融合目标导向强化学习和分层强化学习方法的优势，以应对覆盖组播在目标设定与策略协同上的挑战，是构建高效的覆盖组播优化方法的关键。

通过上述分析，本文提出了一种基于目标导向的分层强化学习（goal-oriented hierarchical reinforcement learning, GO-HRL）方法，通过将覆盖组播树构建任务拆解为若干“源–目的节点对”与对应的“路径规划”。结合分层结构完成“源–目的节点对”和“路径规划”两个层级子任务，上层策略网络负责在当前阶段选择合适的源–目的节点对作为子目标，引导下层策略在全局范围内进行单播路径构建，逐步构建完整的覆盖组播拓扑结构。通过目标导向的层次分工机制，有效应对覆盖组播中策略维度高、子任务耦合紧密等难题，增强了策略在SDN架构下的部署灵活性与对动态网络环境的适应能力。相比现有方法，本文方法GO-HRL在时延、带宽利用率、丢包率等性能指标上均显著优于开放最短路径优先（open shortest path first, OSPF）及主流基于强化学习的方法。本文的主要贡献如下。

1) 针对覆盖组播过度依赖终端节点、缺乏对底层物理网络状态信息实时感知、适应网络状态动态变化能力有限等问题，本文设计了基于目标导向分层强化学习的SDN覆盖组播路由架构GO-HRL，实现对底层物理网络链路带宽、时延、丢包率等网络状态信息的实时感知，为采用RL方法解决覆盖组播策略决策提供全局信息支撑，提升覆盖组播对动态网络环境的适应能力。

2) 针对覆盖组播树求解问题特点设计了高低层策略网络的协同机制。构建融合链路带宽、时延、丢包率以及当前覆盖组播树结构等信息的分层状态空间表示；由高层智能体根据全局状态选择源–目的节点对，低层智能体在满足链路约束条件的前提下完成对应最优路径搜索，两层智能体通过子目标传递形成闭环协作和优化；针对层级任务差异，构建分层动作空间与奖励机制，强化目标导向策略学习，有效抑制环路和策略陷入局部最优，提升训练的稳定性与策略鲁棒性。

3) 结合目标导向强化学习和分层强化学习方法的优势，通过目标导向机制显式引入覆盖组播任务目标，增强策略学习的方向性，使智能体能够更高效地聚焦于优化最终组播性能指标。通过分层学习则将覆盖组播树构建任务分解为“节点对选择”和“路径规划”两个子任务，实现任务的结构化解耦与策略的分层优化，从而降低动作空间维度并提升学习稳定性。

1 相关工作

本节将现有覆盖组播问题的求解方法归纳为三大类：传统优化方法、群智能优化方法和人工智能优化方法，并依次对各类方法进行介绍与分析。

1) 传统优化方法：通常依赖启发式设计、结构优化或预设机制，在路径构建与资源分配中不涉及自学习机制。Chen等[26]提出了任意源容量约束型覆盖组播（any-source capacity-constrained overlay multicast, ACOM）架构，采用“局部随机扩散+环段遍历”的两阶段分发机制，“隐式”地为每个源构建满足能力约束的覆盖组播树。Joung等[27]提出了最短路径覆盖多播（small-group peer-to-peer multicast, SPM）机制，摒弃显式覆盖组播树，直接利用底层单播路径进行数据传输，降低了冗余与控制开销。Shukla等[14]结合SDN技术与分布式哈希表结构，通过逻辑层与物理拓扑的对齐，设计了支持单播与组播的维护触发机制，实现了结构化覆盖网络中路径一致性与维护效率的协同优化。然而，传统优化方法大多基于局部网络状态构建路径，缺乏全局网络状态的统筹规划，在动态大规模网络环境下表现出自适应性差、维护成本高、扩展性不足等问题，难以支撑稳定的覆盖组播通信。

2) 群智能优化方法：通过路径规划和结构优化，具有一定的全局搜索能力和并行性。Ma等[28]提出了基于人工鱼群算法（artificial fish swarm al-

gorithm, AFSA）的覆盖组播路由优化方法，以最小化时延、拉伸度及节点度为优化目标，该方法通过模拟鱼群的随机游动、觅食和追尾等群体行为，结合Pareto排序策略构建高质量覆盖组播树。Gui等[29]设计了基于动态虚拟网格的覆盖组播协议，通过构建和维护虚拟网格结构以适应底层拓扑变化，结合“鱼眼”视图实现了基于局部信息的渐进式自适应更新。Kuo等[30]提出了应用层组播算法ABCD-P2P（advanced bootstrap and adjusted bandwidth for content distribution P2P IPTV），采用双重优化机制：通过高级启动机制，利用两阶段时延测量技术智能引导新节点连接至时延最小的父节点；基于可调带宽机制，根据节点的实际上传能力动态分配子节点数量，并优化节点的位置分布。在上述方法中，尽管群智能优化方法具备一定的全局优化与自适应能力，但普遍存在计算开销大、收敛速度慢、易陷入局部最优，且缺乏对复杂网络状态的全局感知与动态适应能力，难以满足高度动态、异构环境下覆盖组播通信实时性与稳定性要求。

3) 人工智能优化方法：在P2P网络中的应用展现了优化路由、带宽分配与系统调度的巨大潜力。Shoab等[31]提出了基于深度Q学习的查询路由（deep Q-learning based optimal query routing, DRLRS）方法，通过将查询路由建模为深度Q学习任务，利用经验回放和目标网络提高训练稳定性，并引入回报估计机制解决冷启动问题，有效提升了查询效率和网络适应性。Alliche等[32]提出了覆盖层深度Q路由（overlay-deep-Q-routing, O-DQR）框架，采用多智能体深度强化学习结合分布式训练与去中心化执行（distributed training decentralized execution, DTDE）机制，利用logit共享与引导奖励策略，在降低通信开销的同时，有效应对覆盖网络中多跳路由的高动态性与底层不可观测问题，增强了路由的扩展性与稳定性。Nacakli等[33]则通过融合SDN与边缘计算的P2P-CDN架构，采用改进的K-means聚类算法对节点进行分组，在边缘节点集中调度chunk，实现节点分布、调度与路径的统一控制，进而有效减轻内容分发网络（content delivery network, CDN）负载并降低跨域流量。

然而，在覆盖组播问题背景下，直接引入强化学习的相关研究几乎没有。相关领域已有工作展现了潜在思路：刘润滋等[34]提出了基于分层强化学习的中继卫星网络任务动态调度方法，将复杂的调度过程拆解为高低层两个子任务，从而有效降低了状态空间与动作空间的维度，并提升调度效率和任务完成率；Cimurs等[35]提出了基于目标导向深度强化学习的障碍物规避方法，在连续动作空间下实现了目标驱动导航。虽然不直接针对覆盖组播，但这些方法分别体现了分层建模与目标驱动在复杂决策任务中的优势，为本文研究提供了启发。

现有方法虽能取得一些成效，但普遍缺乏面向覆盖组播的有效分层建模与任务解耦机制，未区分全局节点对选择与局部路径规划，导致在多源异构、动态网络中协同效率低、适应性差且易引发策略冲突与训练不稳定。为此，本文在SDN架构下提出了基于目标导向分层强化学习的覆盖组播（goal-oriented hierarchical reinforcement learning overlay multicast, GOHRL-OM）优化方法，通过目标导向任务分解与分层学习，兼顾全局规划与局部路径自适应，实现覆盖组播的高效稳定优化。

2　覆盖组播问题及优化模型

2.1　覆盖组播问题描述

本文所讨论的覆盖组播问题采用SDN架构获取底层网络状态信息，通过在终端节点应用层构建以源节点为根、其余终端节点为叶节点的逻辑覆盖组播树，实现从源节点到多目的节点的可靠数据传输。在网络状态变化时，能够及时调整覆盖组播路径，确保多目的节点间的数据传输效率和稳定性。

网络拓扑表示为无向图$\mathfrak{G}=(\mathcal{V},\mathcal{E})$，其中，$\mathcal{V}$表示有限的节点集合，$\mathcal{E}$表示有限的链路集合。用$v_i \in \mathcal{V}$表示任意网络节点，$i=1,\cdots,n$，$n=|\mathcal{V}|$表示网络节点总数，$e_{ij} \in \mathcal{E}, v_i, v_j \in \mathcal{V}, i \neq j$表示网络节点$v_i$与$v_j$之间的链路，矩阵$\boldsymbol{A}=\left[a_{ij}\right] \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$表示网络的邻接矩阵，其中$a_{ij}$可以表示为

$$a_{ij}=\begin{cases}1, & e_{ij} \in \mathcal{E} \\ 0, & e_{ij} \notin \mathcal{E}\end{cases} \tag{1}$$

假设源节点固定用$v_1^s \subseteq \mathcal{V}$表示，目的节点集合用$\mathcal{V}_d=\left\{v_1^d, v_2^d, \cdots, v_m^d\right\} \subseteq \mathcal{V}$表示，其中，$m=\left|\mathcal{V}_d\right|$表

示所有目的节点个数。

覆盖组播树的构建是一个迭代扩展的过程。该过程从源节点v_1^s向某一目的节点$v_i^d \in \mathcal{V}_d$发送数据。随后，已接收数据的节点和源节点v_1^s均可作为新的源节点，在应用层向目的节点集合$\mathcal{V}_d$中尚未覆盖的节点进行数据分发，直至所有目的节点均被覆盖。最优覆盖组播树以最小化从源节点到所有目的节点的总传输代价。因此，覆盖组播树的每一步构建过程均可分解为两个阶段。1) 节点对选择阶段：确定当前状态下转发数据的源节点和目的节点；2) 路径规划阶段：在当前确定的源–目的节点对之间规划路径并将其加入当前的覆盖组播树结构中。以上覆盖组播树构建过程可以用如下数学形式化方法来描述。

对于包含m个目的节点的覆盖组播任务，需通过m步逐步完成。假设在第k步（其中$k = 0,1,\cdots,m$）中可供选择的源节点候选节点集合用$\mathcal{V}_s^k$表示，可供选择的目的节点集合用$\mathcal{V}_d^k$表示，选定的源节点和目的节点分别可以表示为$v_{i_k}^s \in \mathcal{V}_s^k$和$v_{j_k}^d \in \mathcal{V}_d^k$，求解的覆盖组播树用$T$表示。

初始时源节点集合为$\mathcal{V}_s^0 = \left\{v_1^s\right\}$，目的节点集合为$\mathcal{V}_d^0 = \left\{v_1^d,v_2^d,\cdots,v_m^d\right\}$，$T = \varnothing$。

步骤 1 从$\mathcal{V}_s^0$与$\mathcal{V}_d^0$中分别选取一个源节点$v_{i_1}^s$–目的节点$v_{j_1}^d$组成<源–目的>节点对，满足$v_{i_1}^s = v_1^s$，$v_{j_1}^d \in \mathcal{V}_d^0$，作为首个子目标$\text{subgoal}_1 = \left\langle v_{i_1}^s,v_{j_1}^d \right\rangle$，然后建立规划路径$p_1\left(v_{i_1}^s,v_{j_1}^d\right)$。更新源节点和目的节点集合$\mathcal{V}_s^1 = \mathcal{V}_s^0 \cup \left\{v_{i_1}^s\right\}$，$\mathcal{V}_d^1 = \mathcal{V}_d^0 - \left\{v_{j_1}^d\right\}$，并将路径$p_1\left(v_{i_1}^s,v_{j_1}^d\right)$添加至当前覆盖组播树$T$中。

步骤 2 在更新后的源节点和目的节点集合$\mathcal{V}_s^1$与$\mathcal{V}_d^1$中继续选择<源–目的>节点对，作为下一个子目标$\text{subgoal}_2 = \left\langle v_{i_2}^s,v_{j_2}^d \right\rangle, v_{i_2}^s \in \mathcal{V}_s^1, v_{j_2}^d \in \mathcal{V}_d^1$，然后建立规划路径$p_2\left(v_{i_2}^s,v_{j_2}^d\right)$，更新节点集合$\mathcal{V}_d^2 = \mathcal{V}_d^1 - \left\{v_{j_2}^d\right\}$，并将路径$p_2\left(v_{i_2}^s,v_{j_2}^d\right)$添加至当前覆盖组播树$T$中。

如此不断重复每步过程，当进行至第k步时先确定当前步骤的子目标为$\text{subgoal}_k = \left\langle v_{i_k}^s,v_{j_k}^d \right\rangle$，$v_{i_k}^s \in \mathcal{V}_s^{k-1}, v_{j_k}^d \in \mathcal{V}_d^{k-1}$，然后建立规划路径$p_k\left(v_{i_k}^s,v_{j_k}^d\right)$，再更新节点集合$\mathcal{V}_s^k = \mathcal{V}_s^{k-1} \cup \left\{v_{j_k}^d\right\}$，$\mathcal{V}_d^k = \mathcal{V}_d^{k-1} - \left\{v_{j_k}^d\right\}$，并将路径$p_k\left(v_{i_k}^s,v_{j_k}^d\right)$添加至当前覆盖组播树$T$中。

当目的节点集合$\mathcal{V}_d^k = \varnothing$即$\left|\mathcal{V}_d^k\right| = 0$时，表示所有目的节点均已接收来自初始源节点$v_1^s$发送的数据，覆盖组播树$T$构建完成。为表述方便，假设由路径$p_i,p_j$组成的覆盖组播树为$T(p_i,p_j)$，其中$p_i = \left(V_i,\mathcal{E}_i\right)$，$p_j = \left(V_j,\mathcal{E}_j\right)$，$T(p_i,p_j) = \left(V_T,\mathcal{E}_T\right)$，$V_i$和$\mathcal{E}_i$分别为路径$p_i$的节点集和链路集，$V_j$和$\mathcal{E}_j$分别为路径$p_j$的节点集和链路集，$V_T$和$\mathcal{E}_T$分别为覆盖组播树$T$的节点集和边集。如果本文定义$p_i \oplus p_j \triangleq \left(V_i \cup V_j,\mathcal{E}_i \cup \mathcal{E}_j\right)$，则有$T(p_i,p_j) = p_i \oplus p_j$。由此，最终得到的覆盖组播树$T$可以表示为

$$T = T\left(p_1,p_2,\cdots,p_k\right) = p_1 \oplus p_2 \oplus \cdots \oplus p_k \tag{2}$$

对以上方式构建覆盖组播树性能代价的衡量，参考相关文献[36-37]的做法，本文采用对覆盖组播树中所有路径上性能指标取平均值来反映整体传输性能，对此定义覆盖组播树上路径p_k的性能代价$f\left(p_k\right)$方式如下。

路径p_k的剩余带宽bw表示从源节点v_i^s到目的节点v_j^d的剩余带宽最小值，定义为

$$\text{bw}(p_k) = \min_{e_{ij} \in p_k}\left(\text{bw}_{ij}\right) \tag{3}$$

其中，bw_{ij}为节点i和节点j之间链路的剩余带宽。

路径p_k的总时延delay表示p_k的所有链路时延之和，定义为

$$\text{delay}(p_k) = \sum_{e_{ij} \in p_k} \text{delay}_{ij} \tag{4}$$

其中，delay_{ij}表示节点i和节点j之间链路的时延。

考虑到部分链路的丢包率为0，则路径p_k的丢包率可定义为

$$\text{loss}(p_k) = 1 - \prod_{e_{ij} \in p_k}\left(1 - \text{los s}_{ij}\right) \tag{5}$$

其中，loss_{ij}为节点i和节点j之间链路的丢包率。

覆盖组播树路径p_k的性能代价$f\left(p_k\right)$就是要求最大化剩余带宽$\text{bw}(p_k)$，最小化时延$\text{delay}(p_k)$和丢包率$\text{loss}(p_k)$。其中，剩余带宽反映链路承载能力，时延和丢包率分别表征传输实时性与可靠性。上述3项指标构成多目标优化问题，本文先对其进行Max-Min归一化处理，再采用线性加权转化为单目标优化问题，最终得到覆盖组播树上路径p_k的

性能代价$f(p_k)$，定义为

$$f(p_k)=\beta_1(\mathrm{bw}(p_k)-1)+\beta_2(1-\mathrm{delay}(p_k))+\beta_3(1-\mathrm{loss}(p_k)) \tag{6}$$

其中，$\beta_i, i=1,2,3$分别表示路径带宽、时延和丢包率的权重系数，且满足约束条件$\sum_{i=1}^{3}\beta_i=1$。权重设计旨在综合权衡吞吐能力、实时性与可靠性，通过调整不同权重大小占比有效避免路径拥塞，进而选择低时延和高可靠链路进行传输，从而满足不同应用场景的需求，提升整体路径服务质量。相关权重取值将在实验部分进一步说明。

至此依据前述参考文献[36-37]的做法以及传输性能代价$f(p_k)$的定义，覆盖组播树T代价$C(T)$可表示为对T所有路径p_k代价的平均值，如式(7)所示。

$$C(T)=\frac{1}{m}\sum_{k=0}^{m}\sum_{v_{i_k}^d\in\mathcal{V}_d^k}\sum_{v_{j_k}^s\in\mathcal{V}_s^k}f\left(p_k(v_{i_k}^s,v_{j_k}^d)\right) \tag{7}$$

最优覆盖组播树指覆盖组播树代价$C(T)$最小时的覆盖组播树，假设用T^*表示，如式(8)所示。

$$T^*=\arg\min_T C(T) \tag{8}$$

2.2　问题分解

由以上介绍可知，覆盖组播树本质上是多个单播路径p_k的组合，每一步构建p_k时的<源–目的>节点对是确定的子任务，与目标导向分层强化学习的思路高度契合。在GO-HRL框架下，高层策略负责子目标设定，确定源–目的节点对，低层策略在此基础上完成路径规划。两层策略相互协同，节点对选择决定路径可达性和资源消耗，而路径质量又反馈影响后续节点对选择。因此，本文将覆盖组播树的构建过程划分为两个层级的策略优化任务。

1) 高层策略：负责子目标学习。假设在路径p_k构建时，所有可能的源–目的节点对$\langle v_{i_k}^s,v_{j_k}^d\rangle$组成的集合记为目标空间，可表示为$\mathcal{G}_k=\left\{\langle v_{i_k}^s,v_{j_k}^d\rangle \mid v_{i_k}^s\in\mathcal{V}_s^{k-1},v_{j_k}^d\in\mathcal{V}_d^{k-1}\right\}$，则子目标学习表示根据当前步骤覆盖组播结构与网络状态信息，从目标空间$\mathcal{G}_k$中选择的源–目的节点对$g_k=\langle v_{i_k}^s,v_{j_k}^d\rangle\in\mathcal{G}_k$作为当前的子目标，其优化目标是最小化覆盖组播树代价$C(T)$，具体如式(7)所示。

2) 低层策略：执行建立“路径规划”子任务。在给定子目标$g_k=\langle v_{i_k}^s,v_{j_k}^d\rangle$前提下，从当前源节点$v_{i_k}^s$出发，选择动作$a_t$逐步构建路径，直至到达目的节点$v_{j_k}^d$或达到事先设定最大步数$N$。在此过程中，低层策略的优化目标是最小化路径的传输性能代价$f(p_k)$，具体如式(6)所示。

在每轮路径构建过程中，低层策略通过与环境交互获取即时奖励r_{in}并更新状态s_{t+1}。路径构建完成后，计算路径奖励R_{in}并反馈至高层策略网络计算反馈奖励R_{ex}，用于更新策略并生成下一个子目标g_{k+1}。该高低层协同过程持续迭代，直至覆盖全部目的节点，完成覆盖组播树构建。

综上所述，本文设计了一种基于目标导向分层强化学习算法，通过在分解目标任务与路径执行两个层次的协同学习，实现覆盖组播路径的高效构建与全局性能优化。

3　智能覆盖组播路由系统架构设计

本文结合SDN架构对全局网络状态信息的感知能力与目标导向分层强化学习，设计实现的智能最优覆盖组播构建方法，设计的SDN智能组播路由结构如图1所示，包括数据平面、控制平面与知识平面，各平面的功能职责划分与协同机制叙述如下。

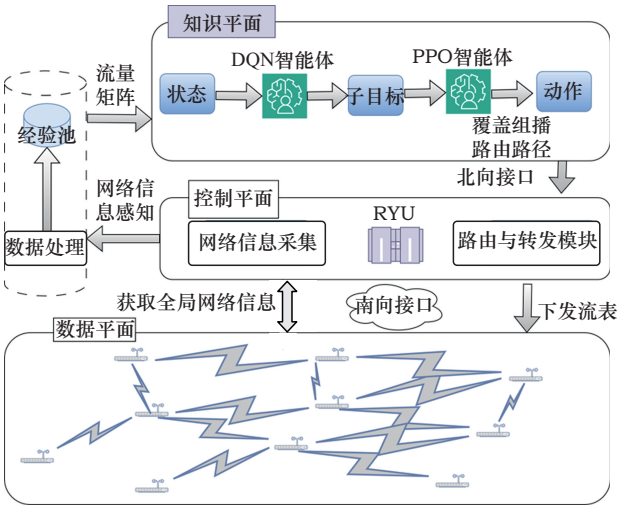


图1　SDN智能组播路由结构

3.1　数据平面

数据平面由网络中的转发设备构成，包括无线接入点（access point, AP）和站点（station, STA）。在本文设计架构下，每个AP下连接一个STA，在

控制平面的统一调度下协同完成覆盖组播的数据转发任务。由于AP通常仅具备数据链路层的数据转发功能，而覆盖组播的实现依赖于STA在应用层对数据进行复制与转发。STA既充当数据接收者，又负责将接收到的数据复制并转发至其他目的STA。AP则通常处于逻辑组播树的中继位置，负责实现多跳转发。此外，数据平面还具备本地链路网络状态的感知能力，能够实时获取全局网络信息（如带宽、时延、丢包率），通过南向接口上传至控制平面，为后续的覆盖组播路由策略调整和全局路径优化提供关键支撑。

3.2 控制平面

控制平面作为网络的“中枢大脑”，集中管理网络资源与控制逻辑。本文采用RYU控制器，通过南向接口基于OpenFlow协议与数据平面进行交互，周期性地从数据平面采集底层网络状态信息与节点运行信息，构建全局网络视图与拓扑模型。

具体而言，控制平面初始采集的网络状态信息包括各端口发送数据包数 tx^{p}、接收数据包数 rx^{p}、发送字节数 tx^{b}、接收字节数 rx^{b}、发送弃包数 $\mathrm{tx}^{\mathrm{drop}}$、接收弃包数 $\mathrm{rx}^{\mathrm{drop}}$、发送错包数 $\mathrm{tx}^{\mathrm{err}}$、接收错包数 $\mathrm{rx}^{\mathrm{err}}$ 以及端口发送字节的持续时间 t^{dur} 等指标。通过在这些端口上采集到的数据进行如下统计计算得到网络链路的性能指标状态信息。

具体而言，链路的剩余带宽 bw_{ij} 定义为链路 p_{ij} 的最大带宽 $\mathrm{bw}_{ij}^{\max}$ 与已使用带宽 $\mathrm{bw}_{ij}^{\mathrm{used}}$ 的差值，通过 tx^{b}、rx^{b} 和 t^{dur} 计算出瞬时吞吐量（即已使用带宽 $\mathrm{bw}_{ij}^{\mathrm{used}}$），如式(9)和式(10)所示。

$$\mathrm{bw}_{ij}^{\mathrm{used}}=\frac{\left|\left(\mathrm{tx}_{j}^{b}+\mathrm{rx}_{j}^{b}\right)-\left(\mathrm{tx}_{i}^{b}+\mathrm{rx}_{i}^{b}\right)\right|}{t_{j}^{\mathrm{dur}}-t_{i}^{\mathrm{dur}}} \tag{9}$$

$$\mathrm{bw}_{ij}=\mathrm{bw}_{ij}^{\max}-\mathrm{bw}_{ij}^{\mathrm{used}} \tag{10}$$

其中，tx_{i}^{b} 和 tx_{j}^{b} 分别表示节点 i 和节点 j 的发送字节数，rx_{i}^{b} 和 rx_{j}^{b} 分别表示节点 i 和节点 j 的接收字节数，t_{i}^{dur} 和 t_{j}^{dur} 分别表示节点 i 和节点 j 的端口发送字节的持续时间。

链路丢包率 loss_{ij} 依据链路 p_{ij} 中源节点 i 的发送数据包数 tx^{p} 与目的节点的接收数据包数 rx^{p} 计算，如式(11)所示。

$$\mathrm{loss}_{ij}=\frac{\mathrm{tx}_{i}^{p}-\mathrm{rx}_{j}^{p}}{\mathrm{tx}_{i}^{p}} \tag{11}$$

链路时延 delay_{ij} 定义为数据在网络链路上传输所需的时间。通过链路层发现协议（link layer discovery protocol, LLDP）报文配合带时间戳的Echo请求进行测量。SDN控制器首先测量从其到源节点和目的节点的往返时延，分别记为 T_{rs} 和 T_{rd}，并记录沿链路正向传播时延 T_{fwd} 与回复传播时延 T_{reply}。其中 T_{fwd} 是控制器到源节点，源节点到目的节点，再返回控制器的传输时延，T_{reply} 是其反向传播的时延，具体如式(12)所示。

$$\mathrm{delay}_{ij}=\frac{T_{\mathrm{fwd}}+T_{\mathrm{reply}}-T_{\mathrm{rs}}-T_{\mathrm{rd}}}{2} \tag{12}$$

3.3 知识平面

知识平面是在SDN架构中新引入的智能决策模块，用于覆盖组播路由路径的构建与优化。在每个调度周期内，知识平面接收来自控制平面的全局网络状态信息，并通过max-min归一化方法统一映射到区间$[0,1]$，如式(13)所示。

$$m_{ij}=\frac{m_{ij}-\min\left(M_{\mathrm{raffic}}\right)}{\max\left(M_{\mathrm{raffic}}\right)-\min\left(M_{\mathrm{raffic}}\right)} \tag{13}$$

其中，m_{ij} 是节点 i 和节点 j 之间的参数矩阵归一化后的元素，$\max\left(M_{\mathrm{raffic}}\right)$ 和 $\min\left(M_{\mathrm{raffic}}\right)$ 分别是参数矩阵的最大值和最小值。

强化学习智能体基于流量矩阵与环境进行交互，通过持续训练优化其策略。在策略收敛后，智能体根据当前网络状态做出高效且稳定的决策，构建覆盖所有目的节点的最优组播路径。最终，构建的覆盖组播树通过北向接口反馈至控制平面，由控制平面下发调度策略至网络设备，完成路径部署与执行。

4 基于目标导向分层强化学习的最优覆盖组播树构建方法

针对最优覆盖组播树问题高维、难以适应网络状态变化等问题，本文设计的目标导向分层强化学习方法GO-HRL，将覆盖组播任务分解为若干个子目标，引入层次化策略结构协同优化源–目的节点对选择与路径规划两个高度耦合的关键子任务：高层策略引导覆盖组播树扩展方向；低层策略在链路约束下完成局部路径规划，实现结构性解耦与协同优化。GOHRL-OM算法架构如图2所示。

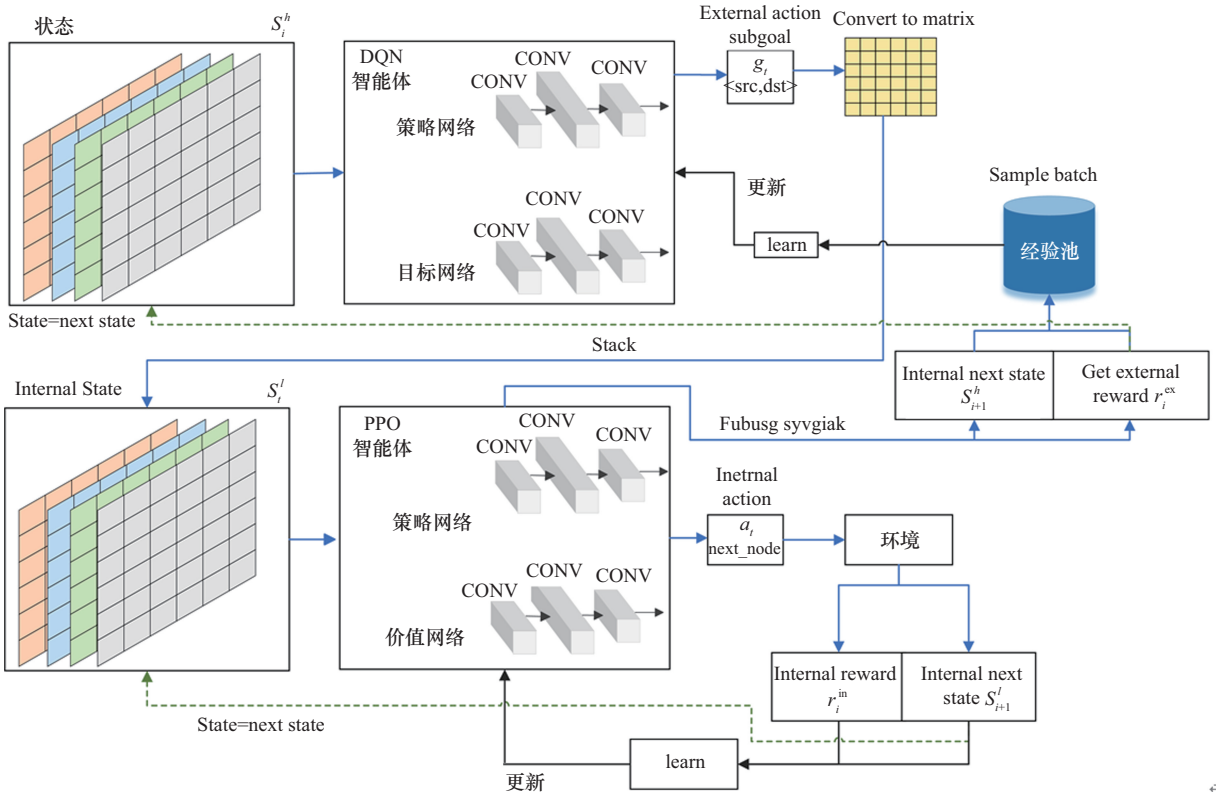


图2　GOHRL-OM 算法架构

相比于传统的强化学习算法架构，基于目标导向的强化学习方法通过扩展的马尔科夫决策过程可表示为一个六元组$\langle S,A,P,R,\mathcal{G},\phi\rangle$，其中，$S$表示状态空间，$A$表示动作空间，$P$表示状态转移函数,$R$表示奖励函数，$\mathcal{G}$表示目标空间，$\phi$表示状态到目标的映射函数。本文设计的高层和低层策略如下。

高层策略采用深度Q网络（deep Q network, DQN）[38]基于当前网络状态对覆盖组播任务进行分解。高层状态输入s_t^h包括当前已构建的覆盖组播树状态矩阵和网络流量矩阵（包含剩余带宽矩阵、链路时延矩阵和链路丢包率矩阵）。高层策略根据当前状态输入s_t^h，通过状态到目标的映射函数$\phi(g_t)$从候选源、目的节点集合中分别选取一个源–目的节点对作为当前阶段的子目标g_t。该子目标g_t经编码形成子目标矩阵，并与网络流量矩阵拼接，构成低层智能体的状态输入s_t^l。

低层策略基于深度近端策略优化（proximal policy optimization, PPO）[39]，围绕当前子目标g_t逐跳规划从源节点到目的节点的路径。在与环境交互过程中，PPO智能体在每一步获得即时奖励R_{part}，并持续更新状态s_t^l，直至完成该子目标。随后计算路径累计奖励R_{in}，更新策略与价值网络的参数，并将子目标完成奖励反馈给高层智能体，高层智能体据此计算对应的高层奖励R_{ex}，并更新自身状态s_{t+1}^h。高层DQN智能体将四元组交互数据$\langle s_t,g_t,R_{\text{ex}},s_{t+1}\rangle$存入经验池，通过批量采样更新网络参数。该过程不断迭代，直至完成所有子目标的选择与路径规划，最终构建出完整且优化的覆盖组播树结构。

综上，GOHRL-OM通过“目标驱动”与“分层协同”融合，实现对覆盖组播任务的高效建模与优化。接下来从状态空间、动作空间和奖励函数3个方面，进一步说明高层和低层智能体的具体设计。

4.1　状态空间

状态空间是智能体对环境的可观测描述。在GO-HRL中，覆盖组播任务被建模为逐步完成多个子目标的过程，状态空间设计围绕“子目标选择与达成”展开，确保决策具有明确的目标导向性。

设计的每一时刻状态空间S由网络流量矩阵$\boldsymbol{M}_{\text{traffic}}$和目标引导矩阵$\boldsymbol{M}_{\text{location}}$两部分组成。网络流量矩阵$\boldsymbol{M}_{\text{traffic}}$是描述节点间链路状态的多通道矩

阵，分别记录链路剩余带宽、时延与丢包率，无连接的位置设为0，表示不可达。该矩阵提供全局网络资源视图。目标引导矩阵 $\boldsymbol{M}_{\text{location}}$ 为对角矩阵，标识当前任务中各节点角色：1表示源节点，−1表示目的节点，0表示未参与节点，作为任务的引导信号。

1) 高层策略网络状态空间设计：用于生成新的子目标，如图3所示。状态 s_t^h 包括网络流量矩阵 $\boldsymbol{M}_{\text{traffic}}$ 与目标引导矩阵 $\boldsymbol{M}^h_{\text{location}}$。高层策略网络在该状态下选择一组源–目的节点对作为子目标。当网络流量矩阵 $\boldsymbol{M}^h_{\text{location}}$ 对角线上仅剩源节点（即所有的目的节点连接完成）时，说明所有目的节点均已被覆盖，覆盖组播任务完成。

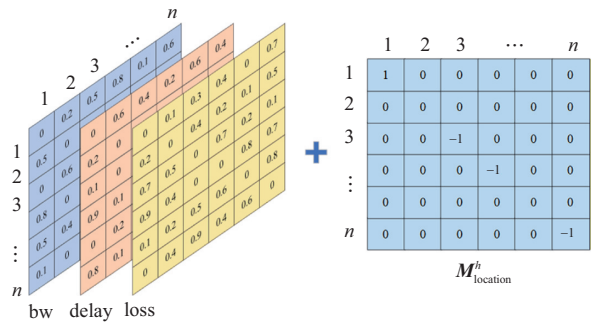


图3 DQN智能体的状态矩阵

2) 低层策略网络状态空间设计：专注于当前子目标的实现，如图4所示。状态 s_t^l 由网络流量矩阵 $\boldsymbol{M}_{\text{traffic}}$ 与仅包含一个源节点与对应目的节点的目标引导矩阵 $\boldsymbol{M}^l_{\text{location}}$ 联合表示。低层智能体在此状态下搜索路径，若路径成功连接源–目的节点，则子目标达成，返回高层策略网络进入下一轮子目标选择。

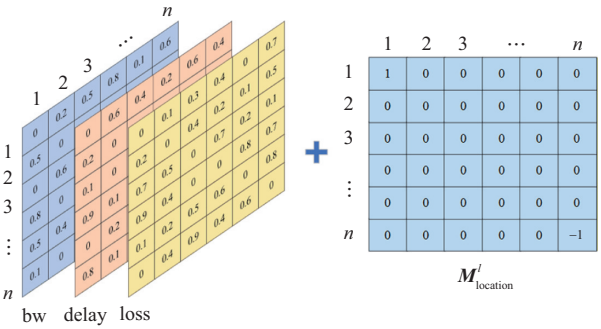


图4 PPO智能体的状态矩阵

这样设计的状态空间形式具备明确的目标引导机制，GO-HRL实现了覆盖组播任务的分层解耦与阶段推进，确保每一步决策聚焦当前目标，提升了整体策略的聚焦性与执行效率。

4.2 动作空间

动作空间是智能体在某一状态可执行的所有有效操作集合。

1) 高层策略网络动作空间设计：从当前源节点与目的节点集合中各选一个节点，组成一个源–目的节点对，作为当前阶段的路径构建目标。

2) 低层策略网络动作选择策略设计：在智能体的每个决策步骤，智能体的候选动作空间由当前智能体所处节点位置的可达邻居节点集合构成。路径始于高层策略选定的源节点，止于对应目的节点。在路径构建过程中，随着末端节点的变化，动作集合动态更新，当智能体选择的下一跳成功到达目的节点时，视为当前子目标完成。

4.3 奖励函数

奖励函数用于衡量智能体动作的价值，引导策略朝向目标优化。在GO-HRL架构中，覆盖组播任务被建模为一系列“子目标选择与达成”，因此，奖励机制也围绕“目标导向”原则设计，分为高层策略网络奖励 R_{ex} 与低层策略网络奖励 R_{in} 两部分，分别用于强化子目标的合理选择与路径规划的质量执行。

1) 高层策略网络奖励 R_{ex} 设计：旨在引导高层智能体高效学习覆盖所有目的节点的最优子目标选择策略。该奖励由低层策略执行子目标路径规划的结果反馈生成，具体设计如式(14)所示。

$$R_{\text{ex}}=\begin{cases}R_{\text{in}}+1, & \text{子目标达成}\\ R_{\text{in}}-1, & \text{子目标未达成}\end{cases} \tag{14}$$

2) 低层策略网络奖励 R_{in} 设计：旨在激励低层智能体高效完成子目标任务，并在路径搜索过程中持续优化路径质量。根据路径搜索过程中的状态转移可能出现的3种情形：过程状态PART、回路状态LOOP和终止状态END。以下分别介绍这3种情况并设计奖励函数。

过程状态PART：当智能体执行某一有效动作，路径新增一个未访问节点并进入非终止状态时，奖励 R_{part} 依据所选链路的剩余带宽 bw_{ij}、时延 delay_{ij} 和丢包率 loss_{ij}，结合权重因子 $\sum_{i=1}^{3}\beta_i=1,\beta_i\in[0,1]$ 进行加权计算，如式(15)所示。该奖励机制通过正向

激励引导智能体优先选择高剩余带宽、低时延和低丢包率的优质链路，从而加速子目标完成进程并保障路径的服务质量。

$$R_{\text{part}} = \beta_1\left(\text{bw}_{ij} - 1\right) + \beta_2\left(\text{delay}_{ij} - 1\right) + \beta_3\left(\text{loss}_{ij} - 1\right) \tag{15}$$

回路状态LOOP：若智能体选择路径中存在已访问过的节点，该行为将导致路径陷入回路，给此类行为一个定值惩罚C_1，以抑制无效探索。

终止状态END：当路径成功连接子目标的源节点与目的节点时，表明路径规划任务完成，给予智能体终止奖励，即内层策略网络奖励R_{in}，奖励值基于整条路径的链路质量综合评估，并作为低层策略网络学习的重要反馈信号，如式(16)所示。

$$R_{\text{in}} = \beta_1\left(\text{bw}_{p_k} - 1\right) + \beta_2\left(\text{delay}_{p_k} - 1\right) + \beta_3\left(\text{loss}_{p_k} - 1\right) \tag{16}$$

4.4　状态到目标的映射函数

在本文方法中，ϕ用于引导高层策略根据状态输入s_t，从候选的目标节点对集合中选择一个源–目的节点对，作为当前阶段的子目标任务，如式(17)所示。

$$g_k = \phi\left(s_t\right) \tag{17}$$

其中，$g_k \in \mathcal{G}_k = \left\{\left\langle v_{i_k}^s, v_{j_k}^d\right\rangle | v_{i_k}^s \in \mathcal{V}_s^{k-1}, v_{j_k}^d \in \mathcal{V}_d^{k-1}\right\}$，$v_{i_k}^s$和$v_{j_k}^s$分别表示备选的源节点和目的节点，$\mathcal{V}_s^{k-1}$和$\mathcal{V}_d^{k-1}$分别表示上一时刻的源节点和目的节点集合，$\mathcal{G}_k$表示目标空间。为设计实现高层策略从状态到目标节点对的映射函数ϕ，本文采用深度Q网络来设计。深度Q网络能够根据当前状态输入s_t，从目标空间中确定需优先覆盖的源–目的节点对作为子目标，从而引导高层策略在复杂网络环境中高效规划覆盖组播树结构，提升覆盖效率与网络稳定性。

4.5　损失函数及网络参数更新方式

本文采用的目标导向分层强化方法[40]，高层策略网络采用DQN算法，低层策略网络采用PPO算法，具体网络更新过程如下。

高层策略网络的目标是在给定状态s_t^h下，学习高层策略π_g，从而选择当前最优子目标g，其对应的最优状态–价值函数如式(18)所示。

$$Q_h^*\left(s_t^h, g\right) = \max_{\pi_g} E\left[R_{\text{ex}}^{t+1} + \gamma \max_{\pi_g} Q_h^*\left(s_{t+1}^h, g_{t+1}\right) \middle| s_t = s_t^h, g_t = g\right] \tag{18}$$

其中，R_{ex}^{t+1}为当前时刻t到下一时刻$t+1$的瞬时奖励，即下层智能体执行子目标规划所获得的累计奖励回报，g_{t+1}为下一刻的子目标，γ为折扣因子。式(18)用于评价当前策略下，状态s_t^h选择的子目标g的价值。因此，高层策略网络损失函数如式(19)所示。

$$L\left(\theta_h\right) = \mathbb{E}_{\left(s_t^h, g, R_{\text{ex}}^{t+1}, s_{t+1}^h\right) \sim D_h}\left[\left(R_{\text{ex}}^{t+1} + \gamma \max_{\pi_{g_{t+1}}} Q_h\left(s_{t+1}^h, g_{t+1}; \theta_h\right) - Q_h\left(s_t^h, g; \theta_h\right)\right)^2\right] \tag{19}$$

其中，θ_h表示上层网络参数，$\left(s_t^h, g, R_{\text{ex}}^{t+1}, s_{t+1}^h\right) \sim D_h$表示从DQN经验池中采样的批量数据。

低层策略网络PPO接收状态s_t^l和上层选定的子目标g，学习低层策略π_l，输出动作a，设计的损失函数如式(20)所示。

$$L\left(\theta_l\right) = \mathbb{E}\left[\min\left(r_t(\theta)\,\text{AF}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\text{AF}_t\right)\right] \tag{20}$$

其中，$r_t(\theta)$为策略更新前后的比值，如式(21)所示，$\pi_\theta\left(a_t | s_t^l, g_t\right)$为策略网络更新后的策略概率分布，$\pi_{\theta_{\text{old}}}\left(a_t | s_t^l, g_t\right)$为更新前的策略概率分布；$\text{AF}_t$为优势函数，其定义如式(22)所示，$V\left(s_t^l\right)$为当前状态价值函数，$V\left(s_{t+1}^l\right)$为下一状态价值函数，$\gamma$为折扣因子；$\text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)$为梯度裁剪函数，用于防止策略更新过快，如式(23)所示，ϵ为裁剪因子。

$$r_t(\theta) = \frac{\pi_\theta\left(a_t | s_t^l, g_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t | s_t^l, g_t\right)} \tag{21}$$

$$\text{AF}_t = R_{\text{in}}^t + \gamma V\left(s_{t+1}^l\right) - V\left(s_t^l\right) \tag{22}$$

$$\text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right) = \begin{cases} 1-\epsilon, r_t(\theta) \leqslant 1-\epsilon \\ 1+\epsilon, r_t(\theta) \geqslant 1-\epsilon \\ r_t(\theta), 1-\epsilon < r_t(\theta) < 1+\epsilon \end{cases} \tag{23}$$

网络参数更新方式：依据当前环境状态s_t^h选取当前最优子目标g_t，作为低层策略网络的任务指引。在该子目标g_t约束下，低层策略网络按时间步逐步执行决策过程。首先，基于当前状态选择动作a_t，与环境交互以获取即时奖励R_{in}^t，根据式(20)计算损失函数并更新低层网络参数。当下层智能体完成子目标或超出规定时间步数限制后，计算反馈奖励R_{ex}^t，同步更新当前状态s_{t+1}^h，并根据式(19)计算损失函数对高层策略网络参数的更新。随后，高层策略网络随即依据更新状态选择新的子目标，并进入下一轮路径规划。通过高低层的循环协同优化，智能体最终学习并收敛于最优的覆盖组播树构建策略。

4.6 分层强化学习算法流程设计

本文设计的GOHRL-OM算法框架实现如算法1所示，目标是从当前网络拓扑$\mathfrak{G}=(\mathcal{V},\mathcal{E})$中寻找从源节点到目的节点集合的最优覆盖组播树$T$，算法输入包括当前的环境状态$s$、DQN智能体学习率$\alpha_1$、每次采样的batch大小$N_1$、PPO智能体学习率$\alpha_2$、截断次数$K$，以及总训练轮数Episode。

第1行至第3行对目标导向分层强化学习框架涉及的所有网络和经验回放池进行初始化；第4行进入主训练的循环；第5行从信息存储池中读取当前网络拓扑$\mathfrak{G}=(\mathcal{V},\mathcal{E})$；第6行至第11行在初始化DQN智能体的学习状态$s_t^h$后，进行子目标$g_t$的选择，并将子目标$g_t$映射为低层PPO策略网络初始状态$s_t^l$；第12行至第22行，PPO智能体基于当前策略$\pi_l$对环境进行探索，并完成算法更新；第23行至第24行判断PPO算法是否完成子目标；第25行至第29行，根据是否完成子目标，计算DQN智能体的奖励信号，更新环境状态s_{t+1}^h，并将$\left(s_t^h,g_t,R_{\text{ex}}^t,s_{t+1}^h\right)$存入经验池，从而更新DQN算法。最终，智能体通过多轮训练，学习得到覆盖组播树T的构建策略。

算法1 GOHRL-OM

输入 网络拓扑$\mathfrak{G}=(\mathcal{V},\mathcal{E})$，当前的环境状态$s$，DQN智能体学习率$\alpha_1$，经验池采样的batch大小$N_1$，PPO智能体学习率$\alpha_2$、截断次数$K$和总训练轮数Episode

输出 从源节点到目的节点集合的最优覆盖组播树T

1) 初始化网络参数w_1,w_2和θ，DQN的网络$Q_{w_1}(s,g)$，PPO的Critic网络$Q_{w_2}(s,a)$和Actor网络$\pi_\theta(s)$，并复制参数$w_1^- \leftarrow w_1$，初始化DQN目标网络$Q_{w_1^-}(s,g)$，DQN经验回放池D_h
2) for $e=1 \rightarrow$ Episode do
3) for $\boldsymbol{M}_{\text{traffic}}$ in网络信息存储池do
4) 根据$\mathcal{V}_s,\mathcal{V}_d$初始化目标引导矩阵$\boldsymbol{M}_{\text{location}}^h$
5) 拼接矩阵$\boldsymbol{M}_{\text{traffic}},\boldsymbol{M}_{\text{location}}^h$得到高层策略网络的初始状态$s_t^h$
6) while true do
7) 根据当前策略选择子目标$g_t=\pi_g(s_t^h)$
8) 执行子目标g_t，并映射下层网络目标引导矩阵M_{location}^l
9) 拼接矩阵$M_{\text{traffic}},M_{\text{location}}^l$得到低层策略网络初始状态$s_t^l$
10) while true do
11) 根据当前策略$a_t=\pi_l(s_t^k)$选择动作
12) 执行动作a_t，获得奖励值R_{in}^t，环境状态变成s_{t+1}^l
13) for训练轮数$k=1 \rightarrow K$ do
14) 根据式(20)更新低层智能体Critic网络和Actor网络参数$Q_{w_2}(s,a)$，Actor网络$\pi_\theta(s)$
15) if s_{t+1}^l到达目的节点
16) 退出当前训练
17) end if
18) end for
19) if完成子目标或训练时间步超过K
20) 退出当前训练
21) end if
22) end while
23) 根据子目标g_t执行的结果，计算奖励R_{ex}^t，更新$\mathcal{V}_s,\mathcal{V}_d$，状态变为$s_{t+1}^h$
24) 将$\left(s_t^h,g_t,R_{\text{ex}}^t,s_{t+1}^h\right)$存放经验池$D_h$
25) if D_h样本数量≥N_1 do
26) 从D_h采样N_1个元组$\left\{\left(s_t^h,g_t,R_{\text{ex}}^t,s_{t+1}^h\right)\right\}_{i=1,\cdots,N_1}$
27) 根据式(19)更新高层智能体网络参数$Q_{w_1}(s,g)$和$Q_{w_1^-}(s,g)$

28)　end if
29)　if $|\mathcal{V}_d| = 0$ do
30)　　完成覆盖组播树构建策略学习
31)　end if
32)　end while
33)　end for
34) end for
35) 智能体学习完成构建覆盖组播树 T 策略

5　实验设置与性能评估

5.1　仿真环境设置

本文基于Mininet-Wi-Fi 2.3.1b仿真环境平台构建网络拓扑，采用Ryu 4.3.4作为SDN控制器。整个仿真环境部署于GeForce RTX 3090显卡的Ubuntu 20.04.6服务器上。为了模拟真实的网络流量，使用Iperf工具在网络节点间互相发送用户数据报协议（user datagram protocol, UDP）数据包。最终，利用Python 3.8与Pytorch 1.11.0实现SDN控制器与强化学习算法之间的交互。

本文的网络拓扑参考文献[41]，设计了3种不同的网络拓扑，分别由10、14和21个无线节点组成，分别命名为Node10Net、Node14Net和Node21Net，用于评估GO-HRL的性能，如图5所示。网络拓扑链路参数均为随机生成的，且符合均匀分布。其中，链路带宽范围为5~40 Mbit/s，时延范围为1~10 ms，AP之间的距离设置在30~120 m。为了更加真实地模拟实际网络环境，本文采用Iperf工具参考文献[24]模拟一天24 h的网络流量变化情况。

5.2　性能指标

本文采用覆盖组播树中各路径的瓶颈剩余带宽、时延和丢包率作为评价指标，如式(24)所示。

$$\begin{aligned}\overline{\mathrm{bw}}_{\mathrm{tree}} &= \mathrm{average}\frac{\sum\limits_{p_k\in\mathrm{tree}}\mathrm{bw}_k}{K}\\ \overline{\mathrm{delay}}_{\mathrm{tree}} &= \mathrm{average}\frac{\sum\limits_{p_k\in\mathrm{tree}}\mathrm{delay}_k}{K}\\ \overline{\mathrm{loss}}_{\mathrm{tree}} &= \mathrm{average}\frac{\sum\limits_{p_k\in\mathrm{tree}}\mathrm{loss}_k}{K}\end{aligned} \tag{24}$$

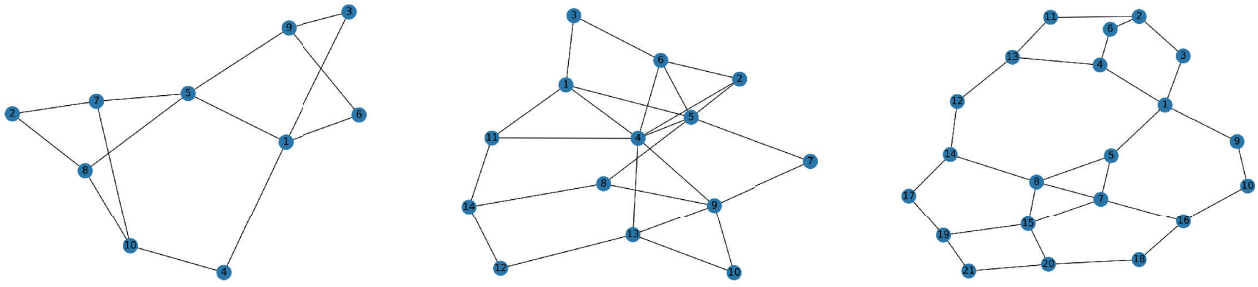
其中，$\overline{\mathrm{bw}}_{\mathrm{tree}}$、$\overline{\mathrm{delay}}_{\mathrm{tree}}$和$\overline{\mathrm{loss}}_{\mathrm{tree}}$分别表示覆盖组播树的瓶颈剩余带宽、时延和丢包率；$\mathrm{bw}_k$、$\mathrm{delay}_k$和$\mathrm{loss}_k$分别表示覆盖组播树中源节点到目的节点路径$p_k$的瓶颈剩余带宽、时延和丢包率，$K$表示$p_k$的数量。

5.3　目标导向分层强化学习方法参数设置

对于路径的性能代价式(6)和智能体奖励机制式(15)中的权重$\beta_i, i = 1,2,3$设置，本文兼顾网络吞吐能力、实时性与可靠性进行权重分配。其中，剩余带宽在提升网络吞吐能力和抑制链路拥塞方面效果显著，因此赋予较高权重；同时保留时延与丢包率的适当占比，以避免过度偏向单一指标。经过多次不同取值的尝试，最终选取带宽、时延与丢包率的权重分别为0.7、0.2和0.1。

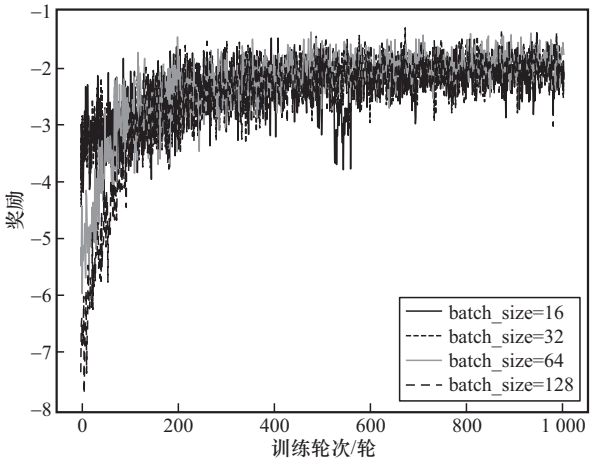
超参数的设置对智能体的性能表现和收敛速度具有重要影响。为提升策略学习效果，分析不同超参数对智能体的影响，从而选择最优超参数。

高层策略网络DQN批量大小batch_size是每次模型训练时采样的样本数量。该参数对模型的收敛速度和最终性能具有显著影响。较小的batch_size引入梯度噪声，增强探索性和多样性，有助于跳出局部最优，但训练不稳定、收敛较慢；较大的batch_size则使梯度估计更稳定、收敛更快，但探索性不足，易陷入局部最优，泛化能力受限。不同batch_size奖励结果如图6所示。

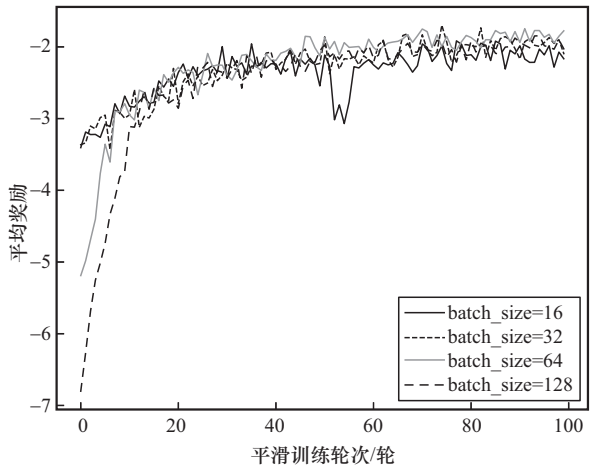


(a) 10个节点　　(b) 14个节点　　(c) 21个节点

图5　无线网络拓扑

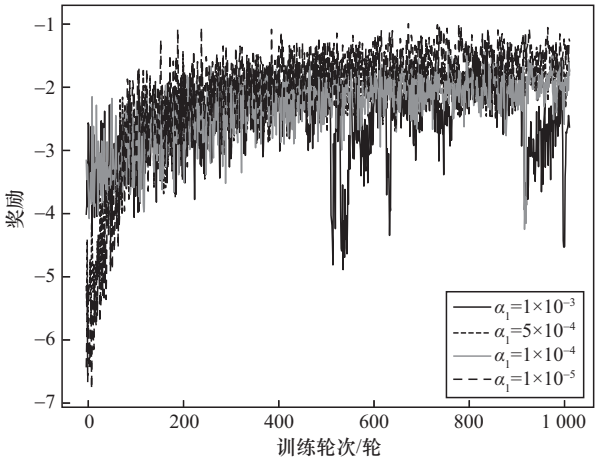


(a) 不同batch_size奖励对比结果

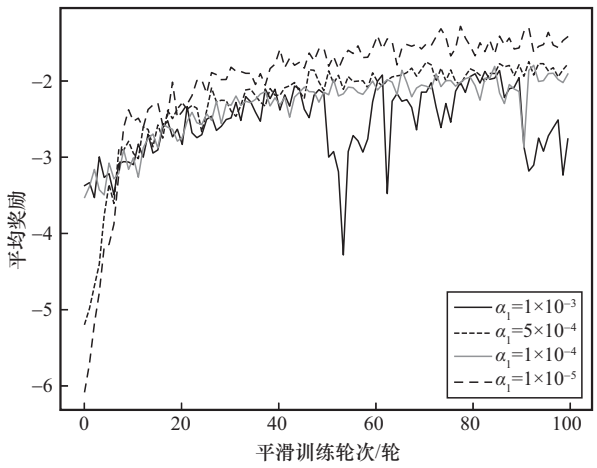


(b) 不同batch_size平均奖励对比结果（窗口大小=10）

图6　不同batch_size奖励结果



(a) 不同学习率α_1奖励对比结果



(b) 不同学习率α_1平均奖励对比结果（窗口大小=10）

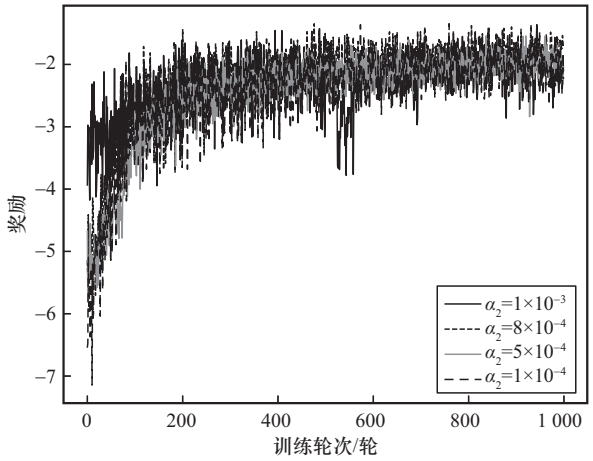
图7　不同学习率α_1奖励结果

实验结果显示，当batch_size=16时，智能体获取的奖励值不能达到收敛；当batch_size=128时，智能体收敛速度较慢，训练轮数在200轮左右时奖励值达到最大值。因此batch_size=64时智能体效果最佳。

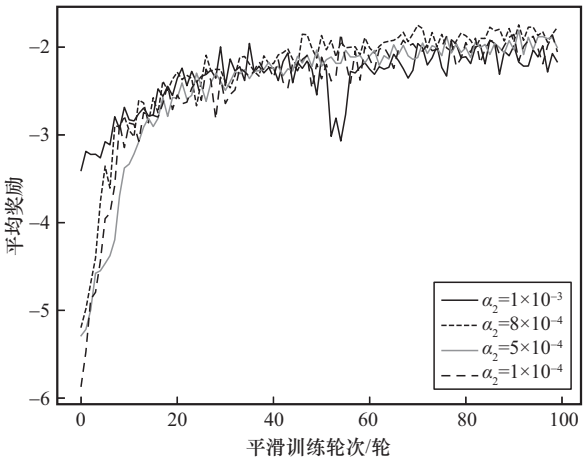
学习率是深度强化学习中的关键超参数，决定模型参数更新步长。学习率过大会导致振荡，甚至不收敛，过小则收敛过慢。本文采用分层强化学习算法框架，包含DQN价值网络以及PPO的Actor和Critic两个神经网络。为分析学习率对模型性能的影响，首先固定PPO中Actor和Critic网络学习率为固定值，然后调整DQN价值网络的学习率进行实验，结果如图7所示。当$\alpha_1 = 1 \times 10^{-3}$时，学习率较大的智能体获取的奖励值难以收敛；当$\alpha_1 = 5 \times 10^{-4}$、$\alpha_1 = 1 \times 10^{-4}$和$\alpha_1 = 1 \times 10^{-5}$时，奖励值可以收敛，但$\alpha_1 = 5 \times 10^{-4}$时收敛速度最快，$\alpha_1 = 1 \times 10^{-5}$时收敛获得的奖励值更大。

固定DQN价值网络学习率$\alpha_1 = 1 \times 10^{-5}$，PPO的Critic网络学习率$\alpha_3 = 1 \times 10^{-4}$，调整PPO的Actor网络学习率$\alpha_2$，结果如图8所示。实验结果表明，当Actor网络学习率$\alpha_2 = 1 \times 10^{-3}$时，学习率较大的智能体获取的奖励值难以收敛；当$\alpha_2 = 8 \times 10^{-4}$、$\alpha_2 = 5 \times 10^{-4}$和$\alpha_2 = 1 \times 10^{-4}$时，奖励值可以收敛，但是$\alpha_2 = 8 \times 10^{-4}$时收敛速度快，并且获得的奖励值更大。

固定DQN价值网络学习率$\alpha_1 = 1 \times 10^{-5}$，PPO的Actor网络学习率$\alpha_2 = 8 \times 10^{-4}$，调整PPO的Critic网络学习率$\alpha_3$，结果如图9所示。实验结果表明，当Critic网络学习率$\alpha_3 = 1 \times 10^{-3}$时，学习率较大的智能体获取的奖励值难以收敛；当$\alpha_3 = 8 \times 10^{-4}$、$\alpha_3 = 4 \times 10^{-4}$和$\alpha_3 = 1 \times 10^{-4}$时，奖励值可以收敛，但是$\alpha_3 = 4 \times 10^{-4}$时收敛速度快，并且获得的奖励值更大。

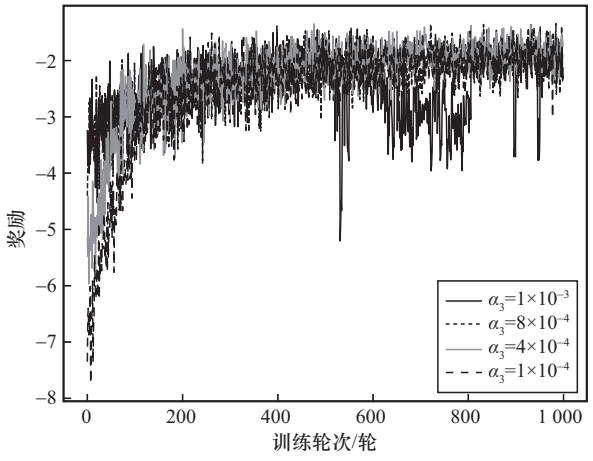


(a) 不同学习率α_2奖励对比结果

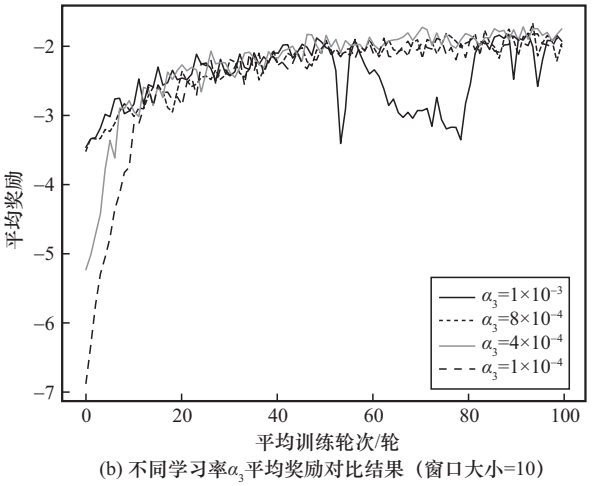


(b) 不同学习率α_2平均奖励对比结果（窗口大小=10）

图8　不同学习率α_2奖励结果



(a) 不同学习率α_3奖励对比结果



(b) 不同学习率α_3平均奖励对比结果（窗口大小=10）

图9　不同学习率α_3奖励对比结果

5.4　对比试验

为了评估GOHRL-OM算法的性能，本文在10、14和21个节点的无线网络拓扑结构中，模拟真实环境的网络流量开展覆盖组播路径优化实验，设置一个源节点与3个目的节点对比算法，包括传统路由协议OSPF[42]、强化学习算法Actor-Critic（AC）[43]和PPO[44]。其中，OSPF作为经典路由方法，至今仍广泛应用于SDN单播与多播研究，并被Mininet仿真平台采用；PPO和AC均属于基于随机策略的在线（On-Policy）强化学习算法，在方法论上与本文提出的GO-HRL框架具有较强的同源性和可比性。由于覆盖组播本质上通过单播实现数据传输，因此本文实验将覆盖组播任务拆解为源节点分别指向目的节点的单播路径进行性能评估，对比指标包括路径的瓶颈带宽、时延和丢包率，以衡量GOHRL-OM算法在多目的地传输任务下的性能表现。

图10为智能体从源节点到各目的节点路径的总链路瓶颈带宽的平均值。实验结果显示，在3个拓扑结构中，相较于传统的OSPF协议，GOHRL-OM算法的平均吞吐量分别提升了6.38%、26.0%和47.18%；相较于AC算法，提升幅度分别为9.04%、28.78%和10.79%；相较于PPO算法，提升幅度分别为41.63%、8.6%和33.49%。结果表明，GOHRL-OM算法能有效规避低带宽链路，提高数据传输性能，满足数据传输时的性能要求。

图11为智能体从源节点到各个目的节点路径总时延的平均值。实验结果显示，在3个拓扑结构中，相较于OSPF，GOHRL-OM算法的平均时延分别降低了5.76%、11.66%和23.15%；相较于AC算法，分别降低了32.8%、44.97%和3.3%；相较于PPO算法，分别降低了55.68%、35.21%和12.74%。上述结果表明，GOHRL-OM算法在路径决策过程中更倾向于选择低时延路径，能够有效降低端到端传输

时延，更好地满足对时延敏感型网络应用的性能需求。

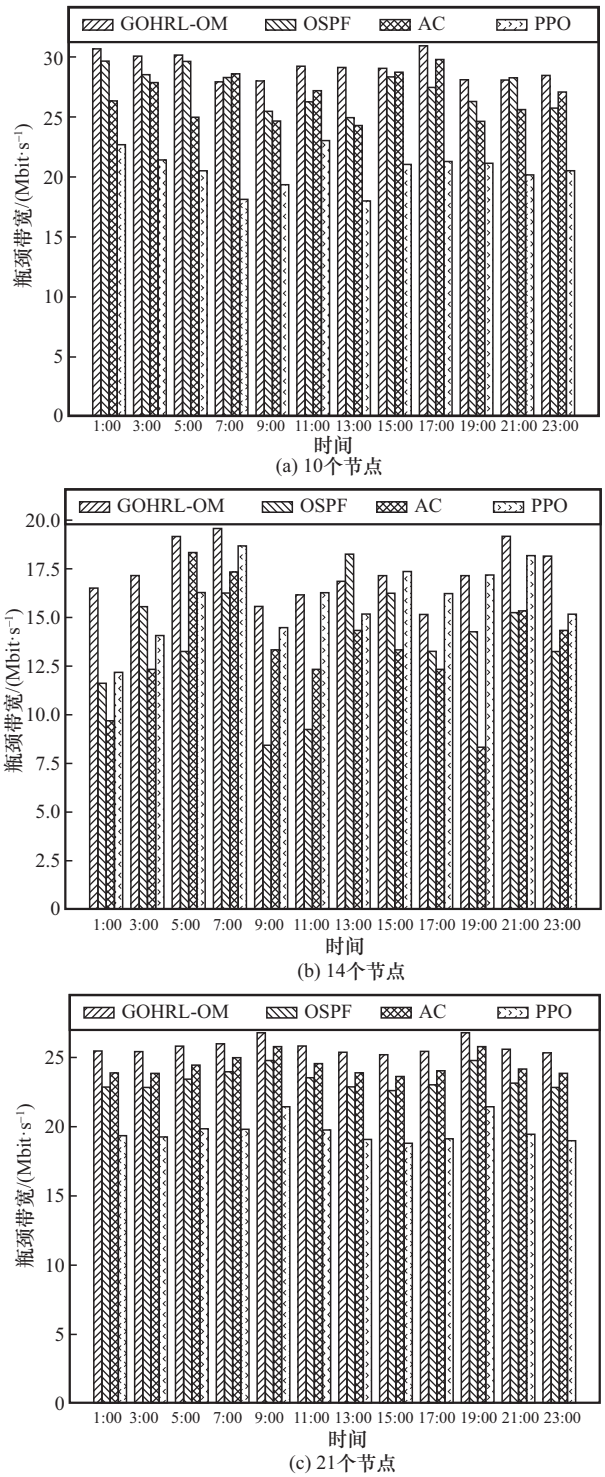


图10　拓扑瓶颈带宽结果

图12为智能体从源节点到各目的节点路径的平均丢包率。实验结果表明，在3个拓扑结构中，相较于OSPF，GOHRL-OM算法的平均性能分别提升了34.41%、33.8%和64.4%；相较于AC算法，分别提升了22.55%、33.54%和32.85%；相较于PPO算法，分别提升了12.55%、28.98%和55.22%。考虑到网络链路在高负载或拥塞状态下存在较高的丢包率，GOHRL-OM算法通过优化路径选择，有效规避拥塞链路，在不同拓扑结构下均显著降低了平均丢包率，从而提升了整体数据传输的稳定性与可靠性。

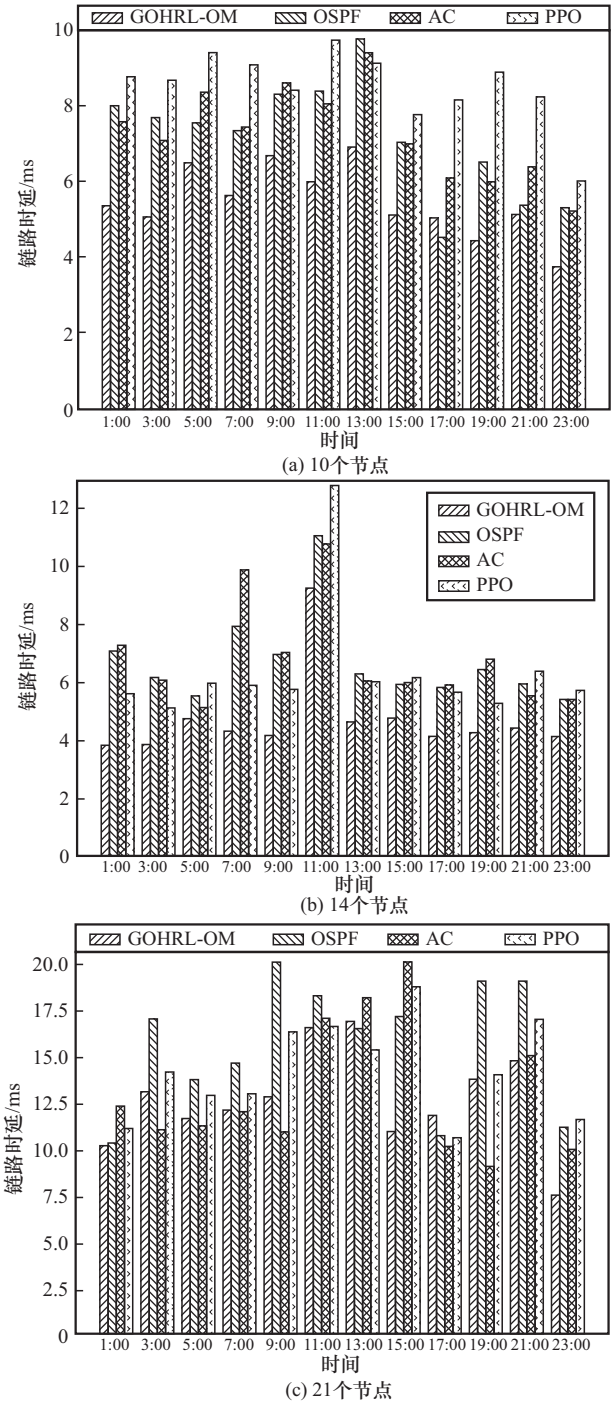


图11　拓扑时延结果

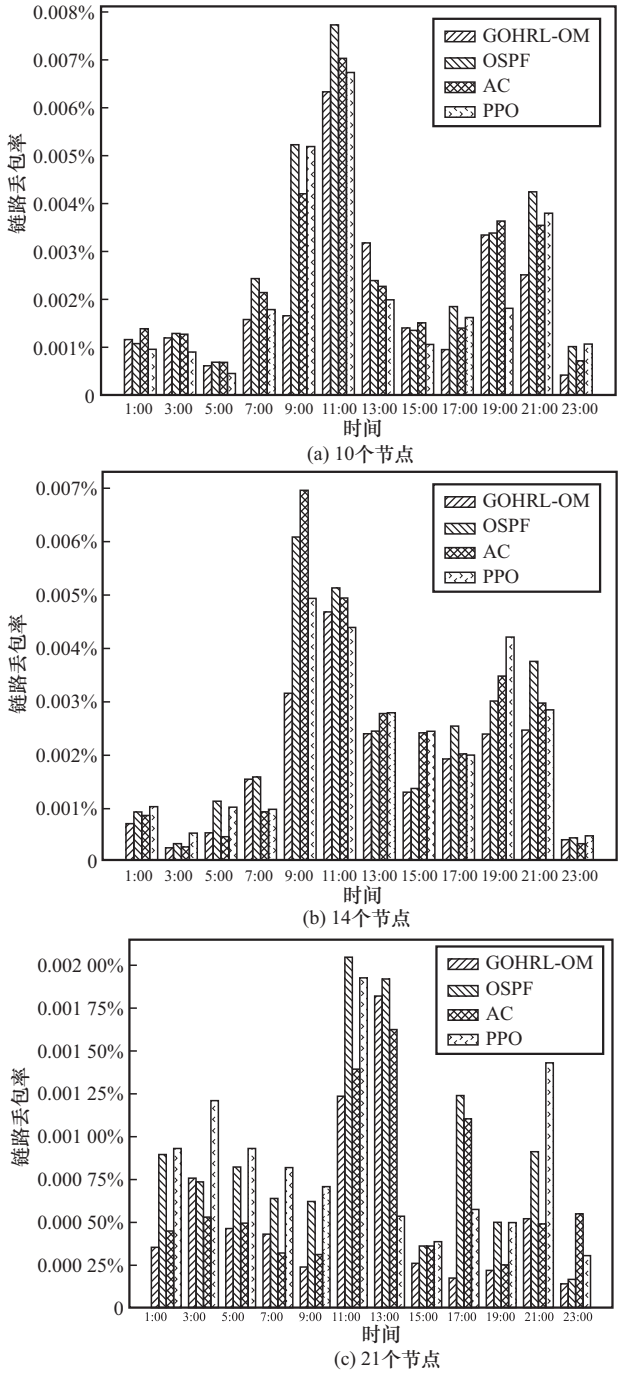


图12　拓扑丢包率结果

6　结束语

本文提出了一种基于目标导向分层强化学习的覆盖组播路由方法GOHRL-OM。首先，针对传统覆盖组播对底层网络状态感知不足、难以适应链路动态变化等问题，通过引入SDN架构，并充分利用控制器的全局观测与可编程能力，实现对链路状态的动态感知与覆盖组播路径的智能调整。其次，针对覆盖组播中存在的高维耦合决策问题，GOHRL-OM结合目标导向机制与分层强化学习框架，将任务划分为节点对选择与路径规划两个阶段，由高低层智能体协同完成路径构建。通过引入子目标机制引导策略学习，能够快速构建满足带宽、时延、丢包率等多约束条件下的高质量覆盖组播树。

实验结果表明，GOHRL-OM在多种网络拓扑结构下均显著优于传统路由协议和主流强化学习算法，在路径吞吐量、时延和丢包率等关键性能指标上表现出更强的稳定性与泛化能力，验证了其在动态复杂网络环境中的实际应用价值。

未来工作将进一步探索在SDN多控制器架构下的覆盖组播路由机制，以适应更大规模网络的管理需求。同时考虑数据平面中STA节点的移动性以及AP节点的动态加入与离开对组播路径的影响，并从计算效率与搜索开销两个方面优化算法性能，以增强其实时性与实用性。

参考文献：

[1] Chiang S H, Wang C H, Yang D N, et al. Online multicast traffic engineering for multi-view videos with view synthesis in SDN[J]. IEEE/ACM Transactions on Networking, 2024, 32(4): 2778-2793.

[2] Zhang X C, Wang Y L, Geng G G, et al. Delay-optimized multicast tree packing in software-defined networks[J]. IEEE Transactions on Services Computing, 2023, 16(1): 261-275.

[3] Smith K D, Jafarpour S, Swami A, et al. Topology inference with multivariate cumulants: the möbius inference algorithm[J]. IEEE/ACM Transactions on Networking, 2022, 30(5): 2102-2116.

[4] Ghosh D, Pandey M, Gautam C, et al. Utilizing continuous time Markov chain for analyzing video-on-demand streaming in multimedia systems[J]. Expert Systems with Applications, 2023, 223: 119857.

[5] Ding Y Q, Wu Z C, Xie L Y. Enabling manageable and secure hybrid P2P-CDN video-on-demand streaming services through coordinating blockchain and zero knowledge[J]. IEEE MultiMedia, 2023, 30(1): 36-51.

[6] Farahani R, Çetinkaya E, Timmerer C, et al. ALIVE: a latency- and cost-aware hybrid P2P-CDN framework for live video streaming[J]. IEEE Transactions on Network and Service Management, 2024, 21(2): 1561-1580.

[7] Qin M, Chen L, Zhao N, et al. Computing and relaying: utilizing mobile edge computing for P2P communications[J]. IEEE Transactions on Vehicular Technology, 2020, 69(2): 1582-1594.

[8] Agarwal V, Ardakanian O, Pal S. Robust peer-to-peer federated learning for non-intrusive load monitoring in smart homes[J]. Energy and Buildings, 2025, 329: 115209.

[9] 崔建群, 陈爱玲, 韩洁, 等. 一种混合的基于分区策略的应用层组播恢复算法[J]. 计算机学报, 2018, 41(9): 1990-2002.
Cui J Q, Chen A L, Han J, et al. A hybrid partition based method for loss

recovery in application layer multicast[J]. Chinese Journal of Computers, 2018, 41(9): 1990-2002.

[10] Hosseini M, Ahmed D T, Shirmohammadi S, et al. A survey of application-layer multicast protocols[J]. IEEE Communications Surveys & Tutorials, 2007, 9(3): 58-74.

[11] 廖怡, 盛益强, 王劲林. 一种基于测量的启发式网络拓扑匹配优化算法[J]. 计算机学报, 2018, 41(9): 2044-2059.
Liao Y, Sheng Y Q, Wang J L. A measurement-based heuristic topology matching optimization algorithm[J]. Chinese Journal of Computers, 2018, 41(9): 2044-2059.

[12] Zhu Y, Li B C, Pu K Q. Dynamic multicast in overlay networks with linear capacity constraints[J]. IEEE Transactions on Parallel and Distributed Systems, 2009, 20(7): 925-939.

[13] Banik S M, Radhakrishnan S, Sekharan C N. Multicast routing with delay and delay variation constraints for collaborative applications on overlay networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2007, 18(3): 421-431.

[14] Shukla N, Datta D, Pandey M, et al. Towards software defined low maintenance structured peer-to-peer overlays[J]. Peer-to-Peer Networking and Applications, 2021, 14(3): 1242-1260.

[15] 叶苗, 胡洪文, 王勇, 等. MA-CDMR: 多域SDWN中一种基于多智能体深度强化学习的智能跨域组播路由方法[J]. 计算机学报, 2025, 48(6): 1417-1442.
Ye M, Hu H W, Wang Y, et al. MA-CDMR: an intelligent cross domain multicast routing method based on multi-agent deep reinforcement learning in SDWN multi controller domain[J]. Chinese Journal of Computers, 2025, 48(6): 1417-1442.

[16] 张克尧, 毕军, 王旸旸. 软件定义网络中基于匹配动作表的IP隧道[J]. 计算机学报, 2019, 42(2): 282-294.
Zhang K Y, Bi J, Wang Y Y. A mechanism of IP tunneling via match-action table in software defined networking[J]. Chinese Journal of Computers, 2019, 42(2): 282-294.

[17] Li Q, Lu L, Zhao D, et al. Stateless and proactive routing for dynamic multicast with deep reinforcement learning[J]. IEEE Transactions on Networking, 2025, 33(5): 2276-2291.

[18] Banerjee S, Kommareddy C, Kar K, et al. OMNI: an efficient overlay multicast infrastructure for real-time applications[J]. Computer Networks, 2006, 50(6): 826-841.

[19] 文鹏, 叶苗, 王勇, 等. SDWN中基于多智能体图强化学习的多对多通信路由方法[J]. 电子学报, 2025, 53(6): 1885-1905.
Wen P, Ye M, Wang Y, et al. A multi-agent graph reinforcement learning method for many-to-many communication routing in SDWN[J]. Acta Electronica Sinica, 2025, 53(6): 1885-1905.

[20] Alhussein O, Zhuang W H. Dynamic topology design of NFV-enabled services using deep reinforcement learning[J]. IEEE Transactions on Cognitive Communications and Networking, 2022, 8(2): 1228-1238.

[21] Huang Q, Yi X Y, Qi F, et al. Enhancing 5G V2X URLLC broadcast/multicast services with FL-based wireless resource allocation[J]. IEEE Transactions on Broadcasting, 2025, 71(2): 384-396.

[22] Hieu N Q, Nguyen D N, Hoang D T, et al. When virtual reality meets rate splitting multiple access: a joint communication and computation approach[J]. IEEE Journal on Selected Areas in Communications, 2023, 41(5): 1536-1548.

[23] Li Y F, Zhang Q, Yao H P, et al. Stigmergy and hierarchical learning for routing optimization in multi-domain collaborative satellite networks[J]. IEEE Journal on Selected Areas in Communications, 2024, 42(5): 1188-1203.

[24] Ye M, Zhao C W, Wen P, et al. DHRL-FNMR: an intelligent multicast routing approach based on deep hierarchical reinforcement learning in SDN[J]. IEEE Transactions on Network and Service Management, 2024, 21(5): 5733-5755.

[25] Leong K Y, Soeung S, Cheab S, et al. Structural optimization for asymmetrical inline topology filter with transmission zeros using goal-oriented reinforcement learning[J]. IEEE Access, 2024, 12: 111386-111399.

[26] Chen S P, Shi B L, Chen S G, et al. ACOM: any-source capacity-constrained overlay multicast in non-DHT P2P networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2007, 18(9): 1188-1201.

[27] Joung U, Jeong H J, Kim D. Overlay small group multicast mechanism for MANET[M]//Personal Wireless Communications. Berlin: Springer, 2006: 205-215.

[28] Ma X, Tang R J, Kang J Y, et al. Optimizing application layer multicast routing via artificial fish swarm algorithm[C]//Proceedings of the 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). Piscataway: IEEE Press, 2016: 115-120.

[29] Gui C, Mohapatra P. Overlay multicast for MANETs using dynamic virtual mesh[J]. Wireless Networks, 2007, 13(1): 77-91.

[30] Kuo J L, Shih C H, Ho C Y, et al. Advanced bootstrap and adjusted bandwidth for content distribution in peer-to-peer live streaming[J]. Peer-to-Peer Networking and Applications, 2015, 8(3): 414-431.

[31] Shoab M, Alotaibi A S. Deep Q-learning based optimal query routing approach for unstructured P2P network[J]. Computers, Materials & Continua, 2022, 70(3): 5765-5781.

[32] Alliche R A, Pardo R A, Sassatelli L. O-DQR: a multi-agent deep reinforcement learning for multihop routing in overlay networks[J]. IEEE Transactions on Network and Service Management, 2025, 22(1): 439-455.

[33] Nacakli S, Tekalp A M. Controlling P2P-CDN live streaming services at SDN-enabled multi-access edge datacenters[J]. IEEE Transactions on Multimedia, 2021, 23: 3805-3816.

[34] 刘润滋, 马天赐, 吴伟华, 等. 基于分层强化学习的中继卫星网络任务动态调度方法[J]. 通信学报, 2023, 44(7): 207-217.
Liu R Z, Ma T C, Wu W H, et al. Dynamic task scheduling method for relay satellite networks based on hierarchical reinforcement learning[J]. Journal on Communications, 2023, 44(7): 207-217.

[35] Cimurs R, Lee J H, Suh I H. Goal-oriented obstacle avoidance with deep reinforcement learning in continuous action space[J]. Electronics, 2020, 9(3): 411.

[36] Hammami C, Jemili I, Gazdar A, et al. HLPSP: a hybrid live P2P streaming protocol[J]. KSII Transactions on Internet and Information Systems (TIIS), 2015, 9(3): 1035-1056.

[37] Mokhtarian K, Jacobsen H A. Minimum-delay multicast algorithms for mesh overlays[J]. IEEE/ACM Transactions on Networking, 2015, 23(3): 973-986.

[38] Llorens-Carrodeguas A, Cervelló-Pastor C, Valera F. DQN-based intelligent controller for multiple edge domains[J]. Journal of Network and

Computer Applications, 2023, 218: 103705.
[39] Wu J W, Zhu Z L. Intelligent routing optimization for SDN based on PPO and GNN[J]. Journal of Network and Computer Applications, 2025, 242: 104249.
[40] 黄志刚, 刘全, 张立华, 等. 深度分层强化学习研究与发展[J]. 软件学报, 2023, 34(2): 733-760.
Huang Z G, Liu Q, Zhang L H, et al. Research and development on deep hierarchical reinforcement learning[J]. Journal of Software, 2023, 34(2): 733-760.
[41] 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.
[42] Li S Y, Wu Q, Wang R, et al. Toward networking and routing in 6G satellite-terrestrial integrated networks: current issues and a potential solution[J]. IEEE Communications Magazine, 2025, 63(3): 92-98.
[43] Nguyen D C, Hosseinalipour S, Love D J, et al. Latency optimization for blockchain-empowered federated learning in multi-server edge computing[J]. IEEE Journal on Selected Areas in Communications, 2022, 40(12): 3373-3390.
[44] Zhou X K, Liang W, Wang K I, et al. Decentralized federated graph learning with lightweight zero trust architecture for next-generation networking security[J]. IEEE Journal on Selected Areas in Communications, 2025, 43(6): 1908-1922.

[作者简介]

叶苗（1977– ），男，广西桂林人，桂林电子科技大学教授、博士生导师，主要研究方向为边缘存储与云存储、软件定义网络、无线传感器网络、模式识别与机器学习。

李繁有（2000– ），男，广西桂林人，桂林电子科技大学博士生，主要研究方向为软件定义网络、强化学习。

文鹏（1994– ），男，贵州毕节人，桂林电子科技大学博士生，主要研究方向为软件定义网络、强化学习、随机优化与应用等。

蒋秋香（1978– ），女，广西桂林人，桂林电子科技大学工程师，主要研究方向为无线传感器网络、人工智能、网络安全。

王勇（1964– ），男，四川成都人，桂林电子科技大学教授、博士生导师，主要研究方向为云计算、网络流量分析与信息安全等。

何倩（1979– ），男，湖南郴州人，桂林电子科技大学教授、博士生导师，主要研究方向为模式识别、机器学习、软件定义网络、传感器网络。

叶聪（2002– ），男，四川成都人，桂林电子科技大学硕士生，主要研究方向为软件定义网络、强化学习。