密 级_____

桂林電子科技大學
GUILIN UNIVERSITY OF ELECTRONIC TECHNOLOGY

硕 士 学 位 论 文

（全日制专业学位硕士）

题目 基于深度强化学习和网络流量状态预测的 SDN 智能路由优化研究

（英文） Research on SDN intelligent routing optimization based on deep reinforcement learning and network traffic state prediction

研 究 生 学 号: 20032303042

研 究 生 姓 名: 黄林强

指导教师姓名、职称: 叶苗 教授

申 请 学 位 类 别: 电子信息硕士

领 域: 计算机技术

论 文 答 辩 日 期: 2023 年 5 月 25 日

独创性（或创新性）声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得桂林电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中做了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：　　日期：

关于论文使用授权的说明

本人完全了解桂林电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属桂林电子科技大学。本人保证毕业离校后，发表论文或使用论文工作成果时署名单位仍然为桂林电子科技大学。学校有权保留送交论文的复印件，允许查阅和借阅论文；学校可以公布论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存论文。（保密的论文在解密后遵守此规定）

本学位论文属于保密在_____年解密后适用本授权书。

本人签名：　　日期：

导师签名：　　日期：

摘 要

随着 5G 网络、云计算和大数据等新技术的发展，网络规模不断扩大，各种新型网络设备不断出现，使得用户对网络服务质量提出了新需求，亟需设计一种高效的智能路由优化方法来保证网络的性能和服务质量。传统网络架构由于控制和转发紧耦合、分散式管理的特点，导致难以获取全局网络状态信息，极大限制了智能路由优化方法的设计与部署，无法满足用户的新需求。软件定义网络是一种新型网络架构，通过南向接口获取网络全局视图信息，北向接口为上层提供应用服务，利用网络开放可编程、控制与数据平面解耦和逻辑上集中控制的优势，实现了对网络的集中统一控制，简化了网络管理，十分有利于智能路由优化方法的设计与部署。本文针对传统路由方法适应动态复杂的网络能力差且无法自适应进行路由转发等缺陷，提出了一种基于 SDN 单控制器管理下的网络智能路由优化方法；同时，考虑在大规模网络中存在 SDN 单控制器负载过大和网络数据包堆积等问题，无法进行实时智能化路由决策，提出一种基于 SDN 多控制器管理下的跨域智能路由优化方法。

本文所做工作及创新总结如下：

（1）SDN 单控制器管理下的网络智能路由优化：针对传统路由方法仅使用有限网络链路状态信息进行路由决策，且适应动态复杂网络变化能力差、调整路由转发策略不灵活的缺陷，本文提出了一种基于 Dueling DQN 强化学习和网络流量状态预测的 SDN 智能路由方法。首先，通过设计的 SDN 多线程网络测量机制获取全局网络感知信息，并将其转换成带宽、时延等多个网络链路状态信息构成的流量矩阵；然后，通过对网络流量矩阵进行预测并使用 Dueling DQN 深度强化学习算法自适应生成当前网络状态下的最佳转发路由。

（2）SDN 多控制器管理下的网络智能路由优化：针对软件定义网络中多控制器域间路由的消息传递和消息同步存在适应时间较长、收敛速度慢，加上传统域间路由方法存在配置繁琐和获取网络状态信息不够灵活等缺陷，导致难以获取全局网络状态信息，不能实时生成最佳路由决策从而影响网络性能，本文提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法。首先，将网络划分成多个子域并由多个本地控制器进行管理，通过设计的 SDN 多线程网络测量机制灵活获取每个网络子域中的状态信息；然后，设计了一种协同通信模块实现根控制器与本地控制器之间的消息传递和消息同步，并通过 socket 技术保证多控制器间消息的可靠性和稳定性传输，实现实时获取全局网络状态信息；最后，由根控制器和本地控制器中的智能体分别自适应生成最优域内和域间路由转发路径后，通过设计一种对网络流量状态进行预测机制以提升跨域智能路由方法的感知能力，实时生成全局网络中最

优路由转发路径。

关键词：软件定义网络；路由优化；多智能体；深度强化学习；网络流量状态预测

Abstract

With the development of new technologies such as 5G network, cloud computing and big data, the network scale has been continuously expanded, and various new network devices have been constantly emerging, which makes users put forward new requirements for network service quality. It is urgent to design an efficient intelligent routing optimization method. Due to the characteristics of tight coupling and decentralized management of control and forwarding, the traditional network architecture makes it difficult to obtain global network state information, which greatly limits the design and deployment of intelligent routing optimization methods and cannot meet the new needs of users. As a new network architecture, the Software-defined Network (SDN) obtains global network state information through the southbound interface, and the northbound interface provides application services for the upper layer. With the advantages of open and programmable network, decoupling control plane and data plane, and logically centralized control, it realizes centralized and unified control of the network, simplifies network management, and is very conducive to the design and deployment of intelligent routing optimization methods. In this paper, a network intelligent routing optimization method based on SDN single controller management is proposed to overcome the shortcomings of traditional routing methods, such as poor adaptability to dynamic and complex networks and inability to adaptively route and forward. At the same time, considering the problems of SDN single controller overload and network packet accumulation in large-scale networks, which can not make real-time intelligent routing decisions, a cross-domain intelligent routing optimization method based on SDN multi-controller management is proposed.

The work and innovation of this paper are summarized as follows:

(1) Network intelligent routing optimization under SDN single controller management: the traditional routing method makes use of limited information on the network links to make routing decisions, which makes it difficult to adapt to the dynamic and complex network and adjust the router's forward strategy. To address these issues, this paper proposes an intelligent routing method based on the SDN, Dueling DQN (a Deep Reinforcement Learning algorithm), and network traffic state prediction. First, the global network awareness information is obtained with the SDN network measurement mechanism, which is converted into a traffic matrix consisting of multiple network link status information such as bandwidth

and delay, etc. Then, the optimal forwarding route under the current network state is generated by predicting the network traffic matrix and the Dueling DQN.

(2) Network intelligent routing optimization under SDN multi-controller management: message transmission and message synchronization for multi-controller interdomain routing in SDN have long adaptation times and slow convergence speeds, coupled with the shortcomings of traditional interdomain routing methods, such as cumbersome configuration and inflexible acquisition of network state information. These drawbacks make it difficult to obtain a global state information of the network, and the optimal routing decision cannot be made in real time, affecting network performance. This paper proposes an SDN cross-domain intelligent routing method based on multi-agent deep reinforcement learning and network traffic state prediction. First, the network is divided into multiple subdomains managed by multiple local controllers, and the state information of each subdomain is flexibly obtained by the designed SDN multithreaded network measurement mechanism. Then, a cooperative communication module is designed to realize message transmission and message synchronization between root and local controllers, and socket technology is used to ensure the reliability and stability of message transmission between multiple controllers to realize the real-time acquisition of global network state information. Finally, after the optimal intradomain and interdomain routing paths are adaptively generated by the agents in the root and local controllers, a prediction mechanism for the network traffic state is designed to improve the awareness of the cross-domain intelligent routing method and enable the generation of the optimal routing paths in the global network in real time.

Keywords: Software-defined network; routing optimization; multi-agents; deep reinforcement learning; network traffic state prediction

目 录

第一章 绪论

§1.1 研究背景与意义

近年来，随着网络规模不断增大，各种新型网络设备不断出现，传统网络架构由于控制和转发紧耦合、分散式管理的特点，不仅不利于网络的更新与维护，而且增加了为网络数据流量提供路由优化服务的难度，无法适应当前网络的需求。软件定义网络（SDN，Software-defined Network）[1][2]的出现为解决上述问题指明了方向。SDN 是一种新型网络架构，通过南向接口（SBI, SouthBound Interface）获取网络全局视图信息，北向接口（NBI, NouthBound Interface）为上层提供应用服务，实现了对网络的集中统一控制，简化了网络管理，降低了网络运营成本，有利于网络更新与部署。与传统 TCP/IP 网络架构相比，SDN 架构具有网络开放可编程、数据与控制平面解耦和逻辑上集中控制等优势。因此，软件定义网络这一技术在诸多方面都优于传统的 TCP/IP 网络架构，能够更好满足当前网络的需求。

在数据中心网络（DCN，Data Center Network）、流量工程（TE, Traffic Engineering）以及服务质量（QoS, Quality of Service）的相关研究中，路由优化一直都是一个重要的研究点。在传统网络架构的发展过程中，已经构建起开放最短路径优先（OSPF，Open Shortest Path First）[3]、路由信息协议（RIP，Routing Information Protocol）[4]、负载均衡（VLB, Valiant Load Balancing）[5]等传统路由方法，在过去的几十年里，这些方法已经被成功地应用到了路由优化等诸多领域。然而，随着网络规模不断增大，网络流量呈现出指数级增长趋势，传统路由方法仅使用有限的链路状态信息进行决策，无法根据 SDN 网络的全局状态信息生成最优路由转发策略，且传统网络架构的路由方法存在收敛速度慢、响应时间长和适应能力差等缺陷，当 SDN 网络规模大的时候这种现象更加明显，显然已经不再适用于 SDN 网络架构。因此，在保证网络服务质量的基本前提下，提出一种新的解决方案来优化 SDN 网络中的路由策略非常重要。

路由优化是提高网络性能，减少网络负载，提升网络服务质量的关键性技术。相较于传统网络测量链路状态信息的方式，基于 SDN 网络架构的优势能够很方便获取网络的全局链路状态信息，达到灵活部署路由策略的目的，因此已经有许多研究人员开展以 SDN 网络架构下的路由优化问题方面的研究。目前，启发式路由算法仍是研究路由优化问题的主要思路。然而，随着人工智能技术的迅速发展，一些智能算法在解决复杂、高维的问题上展现出巨大优势，已经在高量数据处理、复杂策略决策等方面取得了不错的成果。因此，研究 SDN 网络架构中的智能路由优化问题是十分有必要的。

§1.2 国内外研究现状

目前，对于 SDN 网络架构中的路由优化问题的研究，主要从 SDN 单控制器和 SDN 多控制器管理下的网络路由优化两方面展开。

§1.2.1 SDN 单控制器管理下的网络路由优化

基于 SDN 单控制器管理下的网络路由优化问题的研究方法，除了传统的静态和动态路由算法外，目前主流的方法可以分为启发式、机器学习等算法，如图 1-1 所示。

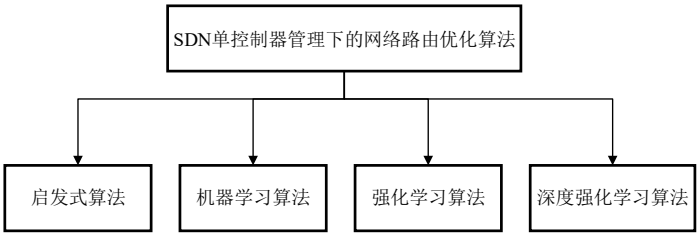


图 1-1 常见 SDN 单控制器管理下的网络路由优化算法

（1）启发式算法

Ahn 等人[6]设计了一种遗传算法，通过求解最短路径来实现最优化路由决策；Derbel 等人[7]考虑到遗传算法容易陷入局部最优解，设计了一种局部搜索的遗传算法来增强搜索空间，从而发现更优的路由转发策略。Zhang 等人[8]针对动态源路径协议（DSR，Dynamic Source Routing）仅以最小跳数进行网络路由转发选择，没有考虑网络节点容量，提出了一种遗传算法-细菌觅食优化算法来进行网络最佳路由选择，首先使用遗传算法快速找到当前网络中较优的路由转发路径，然后通过细菌觅食优化算法来防止遗传算法陷入局部最优解，进而找到网络中全局最优路由转发路径；Parsaei 等人[9]利用 SDN 数控分离的优势，将网络 QoS 建模成一个带约束的线性规划最短路径问题。针对该问题的完全 NP 性，提出了一种基于蚁群算法的求解方法；Jing 等人[10]为了适应电力通信网络不断增大的数据处理和转发需求，提出了一种遗传蚁群路由算法，通过快速搜索得到多个候选最优解，然后利用正反馈方法缩短搜索时间，实现快速获取最优转发路径；Lin 等人[11]基于 SDN 混合网络体系结构下，设计了一种基于 QoS 感知路由的模拟退火算法（SAQR，Simulated Annealing based QoS-aware Routing），根据当前网络状态及预定义的 QoS 需求，实现动态生成路由转发路径；Truong 等人[12]提出了一种基于 SDN 启发式流量工程方法，首先从k条候选路径中选择一条最低成本的路径，然后基于当前网络负载状况使用启发式方法评价当前最低成本路径是否最优，实现了在多径转发中寻找最优转发路径；Ke 等人[13]为了改善无线传感器网络中传感器节点的利用率及数据包路由转发问题，设计了一种人工蜂群算法，

通过 SDN 控制器监控和获取的传感器节点状态信息，动态做出最优路由决策；Shokouhifar 等人[14]针对在虚拟网络中虚拟化路由布局存在 NP 难的现象，设计了一种模糊蚁群优化算法，有效地解决了虚拟网络功能路由布局的难题；Zhang 等人[错误!未找到引用源。]提出了一种粒子群优化算法，用于解决在大规模网络中高效路由部署的难题。首先基于区域划分的思想，将大规模网络划分成多个域，然后通过减少相似数据域间的路由转发，进而提高了信息中心网络（ICN，Information-centric Networking）的路由效率。

以上算法主要采用启发式方式不断迭代来收敛网络状态。然而，此类算法需要有严格的使用场景，网络拓扑和链路状态的变化都可能使启发式算法出现较大波动和误差，导致潜在的可伸缩性问题，进而影响网络的性能。

（2）机器学习算法

Valadarsky 等人[16]使用监督学习来预测未来网络流量变化需求，并针对网络流量变化需求设计了一种监督学习算法以获取最佳路由策略；Sharma 等人[17]设计了一种决策树和神经网络模型，考虑了交换机节点功耗、节点位置及端口流速等因素对模型进行训练，以最小时延寻找网络中数据包的转发路径；Li 等人[18]设计了一种基于多机器学习的路由预设计算法，使用聚类算法来提取网络中流的特征作为模型的输入。接着，使用监督学习预测未来网络流量需求，从而实现对多路径路由自适应策略；Zhou 等人[19]提出了一种基于朴素贝叶斯的路径规划算法，在经过大量数据的学习训练后建立朴素贝叶斯分类器模型，实现了在闭环结构下智能化生成最优传播树的规划线路；Yanjun[20]等人设计了一种基于监督学习的元层框架，在元层中构建了多个机器学习模块，利用启发式算法创建可靠的训练集作为元层的输入，各元层分别得到当前元层的路由策略，最后通过元层汇合得到全局最佳路由策略，有效地提升了网络的性能。Tang 等人[21]利用深度学习端到端的感知能力，设计了一种智能网络流量控制算法，将深度卷积神经网络作为网络主干，降低了无线主干网络中的拥塞程度；针对传统通信系统的路由策略缺乏自适应能力，Mao 等人[22]提出了一种基于深度卷积网络的路由优化方法，通过智能化计算出网络路由转发路径，实现了自适应网络路由转发策略；Kato 等人[23]提出了一个有监督的深度神经网络系统，用于解决深度学习在大规模异构网络下难以建立合适的输入与输出模型的难题，有效提升了大规模异构网络的性能。

上述算法借助机器学习技术，在一定程度上提升了网络的性能。然而，机器学习算法需要获取高量标签数据用于训练网络模型，这将导致极高的计算复杂度。同时，机器学习算法还需要建立一个底层的网络模型，对于复杂、动态的网络而言，很难设计出一种满足所有网络状态需求的模型。

（3）强化学习算法

Hendriks 等人[24]针对 AdHoc 无线网络的路由优化问题，提出了一种基于 Q2-Routing 算法，实现了网络在通信开销和路由质量两方面的权衡；Chen 等人[25]设计了

一种分布式高效缓冲方案，通过基于 TCP 的分布式缓存协议和管理机制，使服务器通过查询缓存表和 Pache 误报信息表来确定发送的数据包，以消除 DCN 中大量冗余的流量，有效提升了网络的性能；Casas-Velasco 等人[26]考虑到传统路由协议对网络流量变化适应的局限性，提出了一种基于 SDN 的 Q-learning Routing 算法，通过获取的链路状态信息做出最优路由决策。Jin 等人[27]设计了一种基于 Q-learning 的路由选路机制，能够满足新型网络中用户多样化的需求，有效降低了 SDN 中业务数据流的丢包率；针对现有卫星路由算法中存在忽略卫星网络状态等缺陷，Yin 等人[28]设计了一种改进的 Q-learning 路由算法，该算法能够提升模型的收敛速度，实现了卫星网络根据网络状态信息动态选择最佳路由的目的。

上述方法是将经验及奖励值以表格形式进行存储，通过 Q-table 查表的方式找到最优路由策略，在一定程度上实现了智能化路由决策的目的。然而，随着网络规模的增大，表格存储空间的大小和查询最优解的时间开销将是巨大的挑战，将导致无法适应动态网络的需求。针对上述问题，许多学者已经使用深度强化学习技术来开展路由优化问题的研究。

（4）深度强化学习算法

Zhao 等人[29]设计了一种基于深度强化学习的智能路由方法，有效缓解了网络拥塞的现象，实现了网络负载均衡；Chen 等人[30]针对复杂、动态网络建模难的问题，提出了一种基于 DDPG（Deep Deterministic Policy Gradient）的深度强化学习算法，将网络分为上下行，并引入多个新特性构成状态空间，动作空间为上下行网络中源-目的交换机之间路径的交集，奖励函数通过调整优化上下行网络的时延和吞吐量，实现了 SDN 流量工程中最优化路由决策；考虑到在传统流量工程中，需要重路由尽可能多的数据流才能保证网络的最佳性能，而频繁重路由会带来网络干扰等问题，Zhang 等人[31]提出了一种基于演员-评论家框架的关键流重路由强化学习算法（CFR-RL，Critical Flow Rerouting-reinforcement Learning）。使用 CFR-RL 算法选择部分关键流进行重路由，大部分流由等价多路径（ECMP，Equal-cost Multi-path）进行转发，有效解决了频繁重路由带来的网络服务质量下降和网络干扰的问题；Fu 等人[32]提出了一种深度 Q-learning 强化学习方法，实现了在数据中心网络中老鼠流低延迟和低丢包率，大象流高吞吐量和低丢包率的目的；Liu 等人[33]设计了一种深度强化学习路由算法，将 SDN 控制器缓存作为影响路由策略的一个关键因素，通过把多个网络资源在减少时延的量化分数来重组缓存和带宽以构成多维状态空间，提高了网络吞吐率和鲁棒性；Hossain 等人[34]设计了一种智能态势感知路由算法，通过智能感知算法来降低网络受到攻击时对应用驱动程序 QoS 造成的影响，有效提升了网络的可靠性；Yu 等人[35]根据深度强化学习机制和黑盒技术的特性，设计了一种可定制、自适应的路由优化策略，有效降低了网络管理和维护成本的开销。

上述算法有效解决了基于 Q-table 方式下的缺陷，利用深度学习与强化学习相结合的技术，加快了模型收敛速度，能够处理和适应复杂、高维的动态网络环境，提升了网络性能。然而，这些方法并没有考虑网络流量状态的未来变化趋势对智能路由优化算法产生的影响，而且一些方法将链路权重值作为动作的输出，还需要进一步计算才能得到路由转发路径。

§1.2.2 SDN 多控制器管理下的网络路由优化

基于 SDN 多控制器管理下的网络路由优化问题的研究方法，目前主流的算法可以分为传统建模优化算法、启发式算法和强化学习算法，如图 1-2 所示。

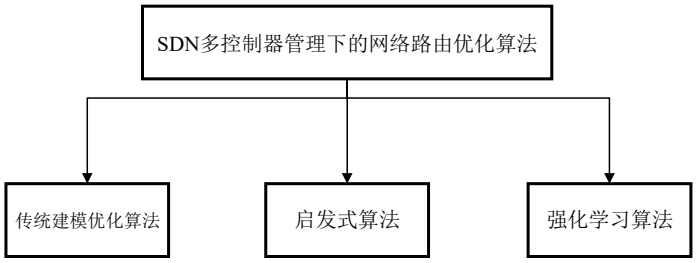


图 1-2 常见 SDN 多控制器管理下的网络路由优化算法

（1）传统建模优化算法

Wang 等人[36]考虑到分布式控制器集群在解决大规模 SDN 网络时，会使本地控制器向根控制器报告域间流而产生响应时间的开销，导致链路负载不均衡，提出了一种 Rounding-Based 算法用于优化 SDN 控制器负载，以更好满足网络的 QoS 需求，有效减少了控制器之间的响应时间，实现了链路负载的平衡。然而，该方法需要构建相关模型和优化目标,而在 SDN 多控制器管理下的大规模网络，构建一个符合多个网络子域的模型十分困难；为了实现分布式 SDN 体系架构中高效路由和低能量消耗，EL-Garoui 等人[37]设计了一种能量感知路由多级映射问题算法（EARMLP），通过能量感知消耗策略寻找控制器-控制器和控制器-交换机之间的路由最优解，提升了网络的性能，该方法同样采取构建相关模型以找出最优解，同样面临优化模型构建难的问题。

（2）启发式算法

Moufakir 等人[38]提出了一种新的多域协作路由框架。首先，设计了一个基于代理的路由框架用于保证网络跨域数据流的性能。然后，提出了一种贪婪算法用于解决大规模网络中多域的路由优化问题，有效提升了网络跨域数据流的整体利用率，降低了网络时延的开销。然而，该方法存在着贪婪算法无法保证获取全局最优解的问题；Xu 等人[39]为了解决 SDN 多控制器的负载不平衡问题，使用控制器集群的负载平衡、平均延迟和交换机迁移成本作为决策因子，设计了一种改进的多目标优化遗传算法（GAIMO），通过局部搜索算子来减少模型的训练时间，用于解决遗传算法中最优解

容易滑向局部最优的问题，有效提升了 SDN 多控制器的总体资源利用率，降低了网络的平均延迟和通信开销，优化了整个网络的性能。然而，该方法需要有严格的使用场景，网络中的拓扑和链路信息发生变化都将使启发式方法出现较大波动和误差，导致多个网络子域之间出现潜在的可伸缩性问题，从而降低网络性能；Zhang 等人[40]提出了一种自适应同步策略用于解决 SDN 多控制器之间同步状态以保证信息的一致性，有效的减少了控制器间的通信开销，提升了网络性能。但是，在网络拓扑改变时，该方法需要花费较长的时间开销以实现自适应同步，从而保证网络全局视图信息的一致性，因此难以满足实时获取全局网络视图的需求；互联网的发展对端到端服务质量（E2E QoS）保证提出了更大需求，尤其是对应用于跨域之间存在的不同类型流量，需要通过较强的稳定性和鲁棒性来协调以保证 E2E QoS，针对此问题 Podili 等人[41]提出了一种信任感知 E2E QoS 路由（TRAQR）框架算法，为多域软件定义网络（MD-SDN）中提供更好的 E2E QoS，有效的保证了网络的质量与性能。该方法同样面临着网络拓扑发生变化时出现的潜在可伸缩性问题，进而影响整个网络的性能。

（3）强化学习算法

Li 等人[42]针对 SDN 多控制器局部过载问题设计了一种基于强化学习的 SDN 多控制器负载均衡机制。通过构建三元组(出迁移域、内迁移域和要迁移的交换机)，在整体迁移效率最佳且无迁移冲突的约束条件下，使用强化学习选择三元组作为当前迭代的最终迁移，以最小代价寻找全局最优解，实现了快速平衡控制器之间的负载，有效降低了迁移的开销成本，从而获得更快的请求包响应速度。然而，该方法面临着如何构建三元组、如何保证在当前三元组条件下整体迁移效率最佳和如何缩短迭代的时间开销等问题。针对传统路由协议仅用有限的信息来实现路由决策，导致对流量可变性的缓慢适应和对应用程序的 QoS 的有限支持，Casas-Velasco 等人[43]提出了一种基于 SDN 环境下的智能路由算法（DRSIR），将可用带宽、低延迟和低损耗的路径优先等因素作为最佳路由选择的度量，以生成适应动态流量变化的智能路由，减少了网络中分组的丢失，降低了网络的时延开销。然而，该方法并未考虑网络流量状态的未来变化趋势对网络性能产生的影响。Wang 等人[44]为了高效的管理 SDN 多域网络中的流，提出了一种协作流管理框架，以最小链路权重为网络中每条流寻找最优路径，有效解决了网络拥塞并显著提升了网络吞吐量，该方法同样未考虑网络流量状态的未来变化趋势对网络性能产生的影响。Godfrey 等人[45]设计了一种多控制器的分层 SDN 架构，提出了一种基于无线传感器网络的 Q-routing 跳数权重调整算法，根据超级控制器获取本地域控制器中的网络信息，在选择下一跳时考虑传感器能量和链路拥塞状态等多个因素，有效提高了无线传感器网络的性能。然而，该方法使用 Q-table 查表的方式寻找最优路由策略，当网络规模增大，表格存储空间的大小和查询最优解的时间开销将是巨大的挑战。Yuan 等[46]提出了一种分布式协同深度强化学习方法，用于解

决车联网的动态控制器分配和控制时延等问题，通过实时分布式协作，本地控制器进行局部决策并与相邻控制器进行协调，有效降低了控制延迟和数据包的丢失。然而，该方法并未考虑动态控制器分配、多控制器之间协作性能和网络状态一致性问题。考虑到启发式迭代网络性能优化算法带来的时间开销问题，Sun 等人[47]设计了一种多智能体的动态控制器负载平衡方法（MARVEL），用于解决在网络流量波动的情况下，控制器-交换机之间的静态映射关系会出现网络负载不均衡，导致 SDN 控制器由于过载而拒绝新的网络流量请求，从而降低控制平面的处理能力，该方法同样没有考虑控制器之间的协同与网络状态一致性的问题。

§1.3 本文主要研究工作

本文针对 SDN 单控制器和 SDN 多控制器管理下的网络路由优化问题展开研究，主要的工作如下：

（1）设计了一种 SDN 多线程网络测量机制。相比传统测量网络链路状态信息的方式，该机制能同时监测网络链路交换机中的端口变化，并获取带宽、时延等链路状态信息，实现灵活获取 SDN 单控制器管理下的全局网络链路状态信息和 SDN 多控制器管理下的每个网络子域中的域内链路状态信息。

（2）提出了一种快速和稳定获取 SDN 多控制器管理下的网络全局状态信息模块。在基于 SDN 多线程网络测量机制的基础上，本文使用所设计的协同通信模块中的 socket 技术进行根控制器与本地控制器之间点对点的通信，实现了域间消息传递与消息同步，提升了 SDN 多控制器管理下的网络中获取全局网络状态信息的收敛速度，并保证了多控制器间消息的可靠性和稳定性传输。

（3）设计并实现了一种基于深度强化学习机制下的 SDN 智能路由优化方法。相比已有文献仅考虑带宽单个网络链路指标构成流量矩阵作为深度强化学习状态空间的设计方法，本文综合考虑了带宽、时延、丢包率、已用带宽等多个网络链路指标，使得智能路由优化算法能够使用多个网络链路指标，依据设计的奖励函数引导智能体进行更好的探索，从而完成从动作空间中选择最优路由转发路径。

（4）在深度强化学习机制下部署了一种网络流量状态预测模型。本文使用 GRU 网络流量状态预测模型，用于发现 SDN 单控制器和 SDN 多控制器管理下的网络中隐藏和未知的流量状态，以提升智能路由优化方法的感知能力。相比常见的 LSTM 预测模型，该模型更加简洁、更加容易收敛。

（5）针对 SDN 单控制器管理下的网络路由优化问题，提出了一种基于 Dueling DQN 强化学习和网络流量状态预测的路由方法。利用所设计的 SDN 多线程网络测量机制实时获取网络的流量矩阵，使用 GRU 网络流量状态预测算法发现隐藏和未知的网络流量状态以提升智能路由方法的感知能力，然后通过深度强化学习中的智能体输

出动作策略，该动作策略直接输出当前时刻网络状态下的最优路由转发路径，实现了自适应智能路由转发操作。

（6）针对 SDN 多控制器管理下的网络路由优化问题，提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法。使用分层式架构将大规模 SDN 网络划分成多个网络子域，通过分布式技术对 SDN 多智能体进行并发学习以减少模型训练时间的开销，利用所设计的 SDN 多线程网络子域测量机制实时获取每个网络子域中的流量矩阵，并通过协同通信模块中的 socket 技术用于多控制器之间点对点的通信，实现域间消息传递与消息同步，提高了获取全局网络状态信息的收敛速度并保证了多控制器之间消息的可靠性和稳定性传输，进而实时获取全局网络中的流量矩阵。最后由本地控制器和根控制器中的智能体分别实时自适应生成当前网络状态下的域内和域间最优路由转发路径后，经多智能体跨域路由算法实时生成全局网络的最优路由转发路径，有效解决了 SDN 单控制器管理下大规模网络中出现负载过大和流量数据包堆积等问题。

§1.4 论文组织结构

本文共由五章内容组成，论文组织结构如下：

第一章介绍了本文的研究背景及意义，分析了当前国内外基于 SDN 单控制器和多控制器管理下的网络路由优化的研究现状及存在问题，并指出本文的主要研究工作。

第二章介绍了相关理论及技术。首先阐述了 SDN 相关技术，然后介绍本文中使用的深度强化学习算法，接着介绍了常见的网络流量状态预测算法。最后给出本文用于评价网络性能三种指标的设计思路。

第三章介绍了本文所设计的基于 SDN 单控制器管理下的网络路由优化方法。针对传统路由优化方法仅使用有限的网络链路状态信息进行路由决策，无法适应动态、复杂的新型网络，难以保证网络的服务质量，提出了一种基于 Dueling DQN 强化学习和网络流量状态预测的 SDN 智能路由方法，并通过相关实验结果及分析，验证了本方法的有效性。

第四章介绍了本文所设计的基于 SDN 多控制器管理下的网络路由优化方法。针对大规模网络中 SDN 单控制器存在负载过大等缺陷，无法进行实时的智能化路由决策，导致出现网络数据包堆积，进而影响网络的性能，提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法，并通过相关实验结果及分析，验证了本方法的有效性。

第五章总结本文的研究工作，并对下一步研究工作进行了展望。

第二章 相关理论及技术

本章将对 SDN 相关技术、深度强化学习算法、网络流量状态预测算法和网络性能评价指标的设计思路和相关理论进行介绍。

§ 2.1 SDN 相关技术

§ 2.1.1 SDN 网络架构

随着网络规模不断增大，各种新型网络设备不断出现，网络流量呈现出指数级增长趋势，基于传统分布式网络架构存在配置繁琐、管理复杂和扩展性差等缺陷，导致不能适应新型网络的需求，难以保证网络的 QoS。SDN 作为一种新型的网络架构，通过网络开放可编程的特点有效解决了传统网络架构扩展性差的缺陷，通过逻辑上集中控制的特点实现了对网络集中管理和集中配置，有效降低了管理网络设备的成本开销，通过数据平面与逻辑平面分离的特点使网络设备只需按照 SDN 控制器下发的策略执行数据包转发操作，实现了对网络流量的细粒化管理，极大提升了网络设备的转发效率。因此，SDN 的出现为适应新型网络需求和保证网络服务质量，提供了一种重要的技术。

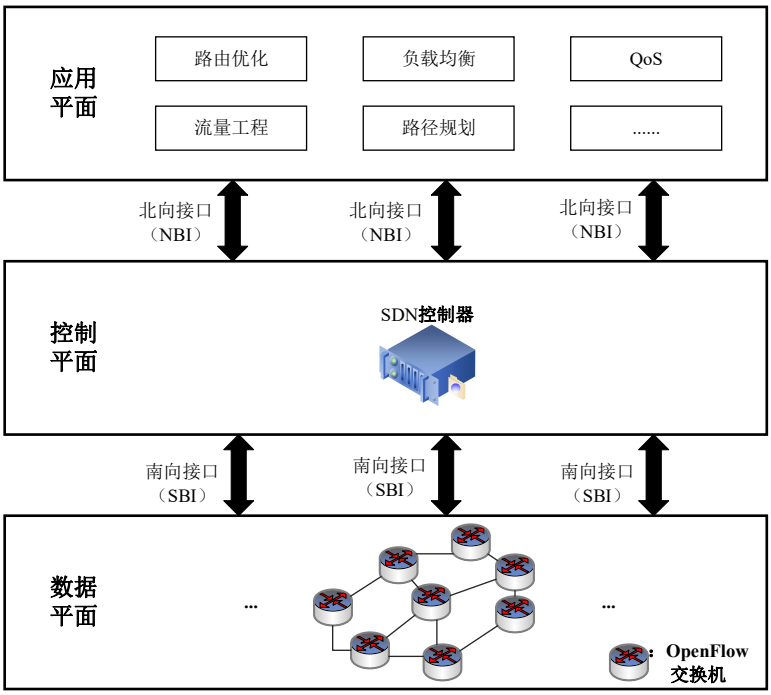


图 2-1 SDN 系统架构

如图 2-1 所示，SDN 系统架构主要由应用、控制和数据三个平面组成。应用平面为用户提供各种应用业务的开发接口，例如：路由优化、流量工程等。控制平面是 SDN 架构的核心组件，主要由 SDN 控制器构成。控制平面以集中控制的方式获取整个网络的视图，使用主动和被动两种模式收集数据平面的拓扑信息，使用北向接口为应用平面提供上层服务，通过南向接口指导数据平面的流表转发操作。数据平面主要由支持 OpenFlow 协议的物理和虚拟交换机组成。它根据控制器下发的流表规则，执行数据包转发操作。

§2.1.2 OpenFlow 协议

OpenFlow 是一种被广泛用于 SDN 控制器和 SDN 交换机之间进行南向接口通信的新型网络协议。自 2009 年底由 ONF（Open Networking Foundation）组织发布 1.0 版本后，OpenFlow 协议已经经过 5 次升级，目前常用的是 OpenFlow1.3 版本。如图 2-2 所示，OpenFlow 架构主要由 OpenFlow 交换机、网络虚拟化层和 SDN 控制器组成。

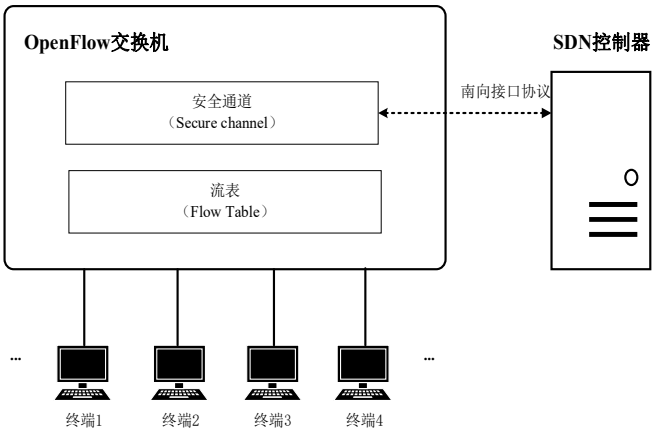


图 2-2 OpenFlow 架构

OpenFlow 交换机主要由流表（Flow Table）和安全通道（Security Channel）两部分组成。其中，流表是用于存放底层设备的转发规则，负责对数据包进行转发操作，由控制器下发的多个流表项构成。图 2-3 为 OpenFlow1.3 协议中的流表项结构,其中优先级：用于在匹配域发生冲突时，选择流表中优先级最高的流表项；匹配域：用于匹配当前数据报文，匹配数据报文的流程如图 2-4 所示；指令：用于指引数据包采取相应的动作，主要包括必备和可选动作；计时器：用于统计数据包匹配的次数；超时：表示流表中当前流表项的有效时间；Cookies：用于过滤旧的流表项。安全通道用于确保 OpenFlow 交换机与 SDN 控制器之间的可靠通信。

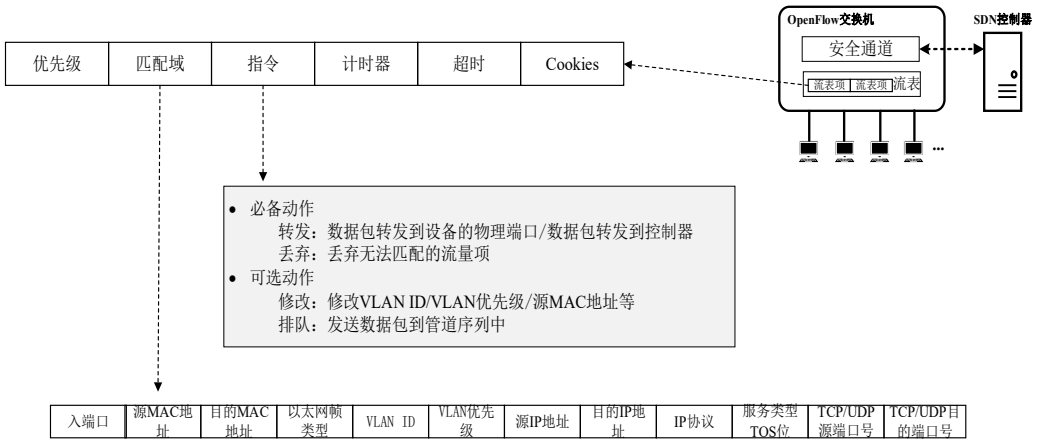


图 2-3 流表项结构

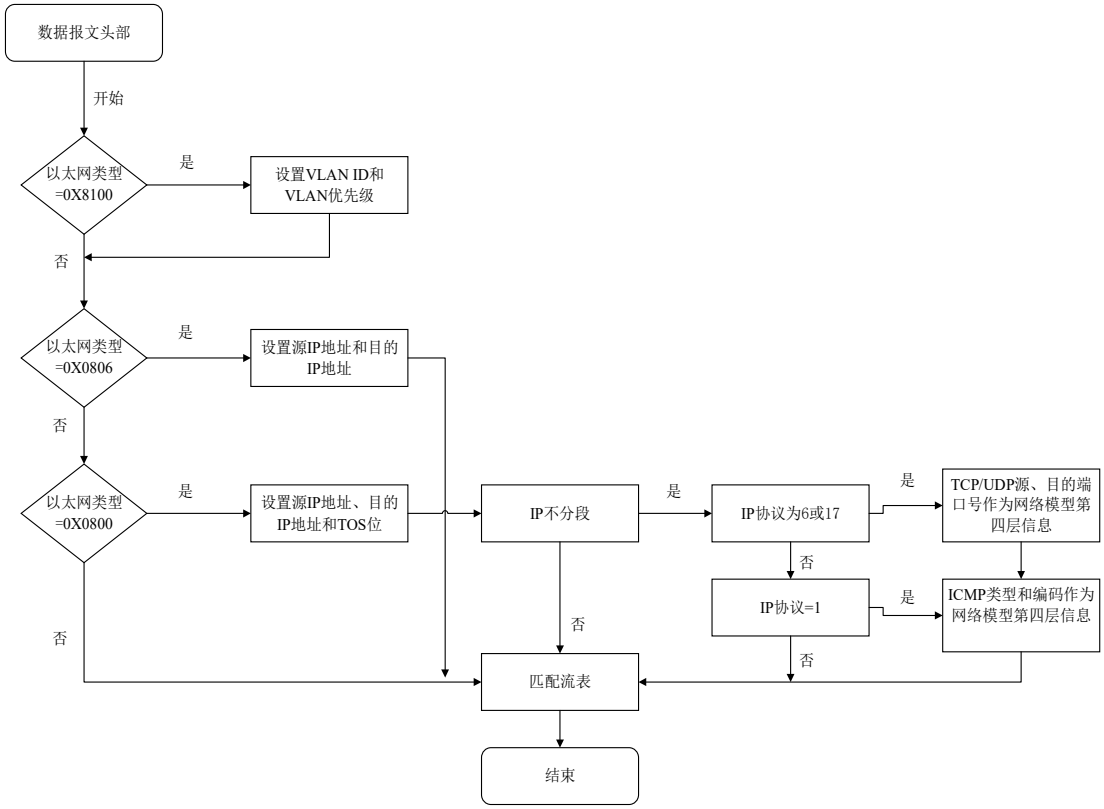


图 2-4 流表项中匹配域的流程图

网络虚拟化层是将物理网络抽象为多个虚拟网络，使用虚拟网络标识符（VNI，Virtual Network Identifier）来标识不同的虚拟网络。虚拟网络之间实现了隔离和互通，因此，提高了网络的可扩展性和灵活性，同时降低了网络管理的复杂度和成本。

SDN 控制器作为整个 SDN 系统架构的中枢，主要负责收集数据平面的全局视图信息、下发相应策略引导数据平面进行转发操作和为应用平面中的业务提供可编程的接口等。在 SDN 控制器与 OpenFlow 交换机的通信中，主要有三种消息类型：控制器到交换机信息（Controller-to-Switch）、异步消息（Asynchronous）和同步消息（Symmetric）三种类型。其中，Controller-to-Switch 消息是由控制器主动向交换机发

送的消息，用于查询交换机状态或下发指令等，主要消息体类型如表 2-1 所示。Asynchronous 消息由交换机向控制器发送的消息，用于通知控制器有关网络状态的变化，例如端口状态的变化、链路故障的发生等等，主要的消息体类型如表 2-2 所示。Symmetric 信息是由控制器和交换机之间相互发送的消息，一般用于交换机在等待一段时间内没有收到控制器的查询数据包时发起，主要的消息体类型如表 2-3 所示。

表 2-1 Controller-to-Switch 消息主要消息体类型

消息体名称	功能
Features	控制器通过向交换机发送 Features 消息，用于请求交换机身份和状态信息，交换机必须对此类消息做出应答处理。Feature 消息通常在安全通道建立时执行
Configuration	用于查询或设置交换机上的配置信息
Modify-State	用于管理交换机流表项和端口状态等，例如：添加、删除或修改流表项
Read-State	用于收集交换机上各种消息，例如：配置、统计等信息
Packet-Out	用于控制器将数据包按交换机指定端口进行转发
Barrier	用于控制器查询当前数据包是否执行转发或丢弃操作

表 2-2 Asynchronous 消息主要消息体类型

消息体名称	功能
Packet-In	交换机将无法匹配流表的数据包 Packet-In 给控制器。当交换机有足够的缓存空间，数据包将临时存放在缓存中；否则，直接发送给控制器
Flow-Removed	交换机中的流表由于超时或者修改等原因被删除时，触发此消息
Port-Status	交换机端口状态发生变化时，触发此消息
Error	交换机发生错误时，触发此消息

表 2-3 Symmetric 消息主要消息体类型

消息体名称	功能
Hello	用于在交换机和控制器之间建立连接
Echo Request/Reply	用于测量网络链路时延
Experimenter	用于为 OpenFlow 交换机提供额外的功能，例如：自定义数据包

§2.1.3 Ryu 控制器

常用 SDN 控制器的性能对比如表 2-4 所示，本文中 SDN 控制器为 Ryu，它是一款由 Python 语言开发、基于开源的控制器，能够适配 OpenFlow 所有版本协议。Ryu 拥有一系列完备的接口、库函数和组件提供给开发者使用。因此，对开发者而言十分友好，开发相关应用业务只需在 Ryu 提供的代码框架上进行编写。Ryu 代码框架如图 2-5 所示，其中 app：用于存放运行的应用业务和为用户提供开发相关应用业务的功能；base：为 Ryu 代码框架提供必要的基类，例如：app_manager.py 用于加载应用程序；cmd：表示入口函数，定义了 Ryu 的命令系统；contrib：用于存放开放社区贡献者的代码；controller：用于处理 OpenFlow 协议的一系列文件，例如：ofp_handler.py 用于建立控制器和交换机之间的对话；lib：用于存放网络相关协议，例如：ICMP、DHCP 协议等；ofproto：用于存放各种 OpenFlow 协议版本的信息；tests：用于提供单元和集成测试功能，测试 Ryu 各模块是否能正常运行；topology：用于查询网络拓扑状态、监听网络事件和获取交换机状态信息等，例如：switches.py 定义了主机、端口和交换机等类。

表 2-4 常用 SDN 控制器的性能对比

控制器	语言	平台	特点
Nox	C++	Linux	开发和部署较为复杂
OpenDaylight	Java	Win/Linux	可扩展、支持多协议
Floodlight	Java	Win/Linux/Mac	跨平台、扩展性强、高稳定性
Huawei eSight	Java	Win	具备自动化管理、监控和故障定位
Pox	Python	Linux	基于 Nox 控制器的扩展、易开发
Ryu	Python	Linux	支持多线程、轻量级、支持 OpenFlow 所有版本协议

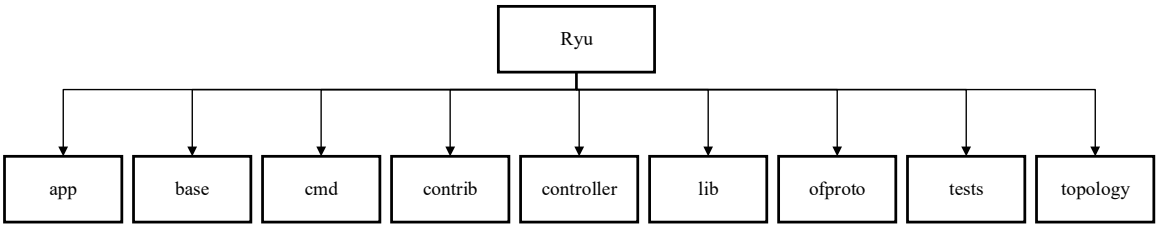


图 2-5 Ryu 代码框架

§2.2 深度强化学习算法

强化学习（RL, Reinforcement Learning）[48][49]是机器学习中的一种范式和方法论。它根据智能体（Agent）在与环境持续交互的过程中，通过学习相应策略获取最大化回报，从而实现最优化的决策目标，已经被广泛应用于自动驾驶、流量工程和路由优化等领域的研究。强化学习常见的模型是标准马尔可夫决策过程（MDP，Markov Decision Process）[50]，主要由状态空间、动作空间和奖励函数三部分组成，如图 2-6 所示。首先,环境生成一个初始状态s_t，智能体根据状态s_t执行相应动作a_t后，获得奖励值r_t及下一个状态s_{t+1}。随后，重复执行上述训练过程，使智能体不断朝着更大奖励值的方向进行学习，从而达到最优化决策的目的。然而，传统的 RL 算法，如基于表格型的 Q-learning[51]和 Sarsa[52]算法，由于存在 Q-table 存储空间和寻找最优解查询时间的限制，很难解决复杂网络中高维状态和动作空间的问题。因此，为了解决以上问题，深度强化学习技术应运而生。

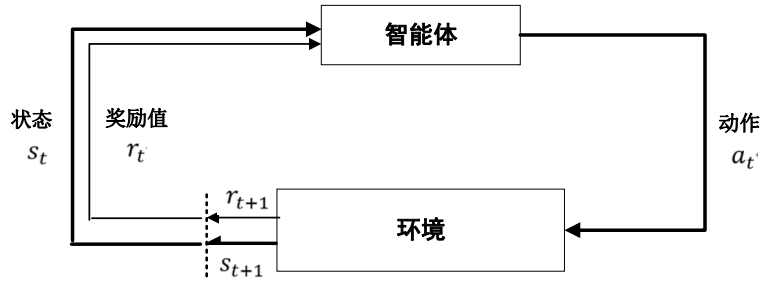


图 2-6 典型强化学习机制

深度强化学习（DRL，Deep Reinforcement Learning）[53]是将深度学习技术（DL，Deep Learning）[54]与强化学习技术相结合，是目前最流行的一种人工智能学习方法。利用深度学习端到端感知和自动提取特征的能力，以及强化学习探索与利用的决策能力，可以有效解决复杂网络中高维状态和动作空间的问题。因此，许多学者已将深度强化学习算法应用到路由优化问题的研究中。如图 2-7 所示，常见的深度强化学习算法可以分成基于无模型（Model-Free）和有模型（Model-Based）的两种类型。

基于有模型的 DRL 算法是根据真实环境设计学习模型，智能体无需与环境交互即可知道任何状态下执行任何动作所获取的收益回报值，甚至下一个状态也能够通过设计的模型计算得出。因此，有模型的 DRL 算法可基于贪心策略，通过在每次决策时选择当前状态下奖励回报最大的动作，即可获得全局最优策略，有效提高了采样效率并减少了决策时间的开销。常见的有模型 DRL 算法有 AlphaZero、MBMF 等。

然而，在现实生活中，很多决策优化问题都难以在真实环境中设计出一种合适的学习模型。此外，在大多数情况下，凭借 DRL 的优势无需对环境进行建模也能获得

最优策略。因此，基于无模型的 DRL 算法更加适用于解决复杂的决策优化问题。无模型的 DRL 算法通过探索和利用机制，使智能体与环境不断进行交互以获取经验样本，从而达到全局最优决策。常见无模型的 DRL 算法可以分成基于价值和基于策略两种类型。下面将介绍本文所使用竞争深度 Q 网络（Dueling DQN，Dueling Deep Q-network）[55]、深度确定性策略梯度算法（DDPG，Deep Deterministic Policy Gradient）[56]和近端策略优化算法（PPO，Proximal Policy Optimization）[57]三种基于无模型的 DRL 算法。其中，Dueling DQN 是本文智能路由优化方法中所使用的深度强化学习算法，而 DDPG 和 PPO 是用于对比的两种深度强化学习算法。

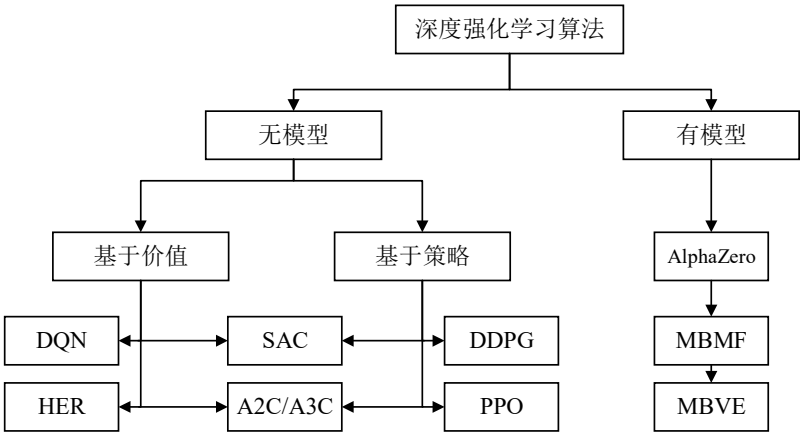


图 2-7 常见的深度强化学习算法

§2.2.1 Dueling DQN 算法

Dueling DQN 算法是基于价值的 DQN 算法改进形式，它能较好改善 DQN 带来的值函数过估计问题，能够提升模型的学习性能，加快模型的收敛速度。与 DQN 算法相对，Dueling DQN 将深 Q 网络中的价值函数分为两部分，第一部分称为价值函数（Value Function）仅与状态s有关，记为$VF(s,\omega,\beta)$；第二部分称为优势函数(Advantage Function)与状态s和具体采取的动作a都有关，记为$AF(s,a,\omega,\beta)$，如公式(2-1)所示：

$$Q(s,a,\beta,\lambda)=VF(s,\omega,\beta)+AF(s,a,\omega,\lambda) \tag{2-1}$$

其中，ω是两部分共有的网络参数，β,λ分别是AF和VF专有的网络参数。Dueling DQN 沿用了 DQN 经验回放和探索机制的优势。如图 2-8 所示，经验回放机制是智能体按一定策略π与环境持续交互，将获取的经验以(s_t,a_t,r_t,s_{t+1})的形式存储在回放缓冲区（M，Replay Memory）中。然后，随机批次采样缓冲区中的数据进行线下训练用于更新网络参数。利用经验回放机制，一方面能够使得智能体重复利用优势经验样本进行高效的经验采样，减少模型在获取经验样本的时间开销；另一方面来自不同策略的经验样本可以降低数据之间的相关性，从而改善模型的学习能力。

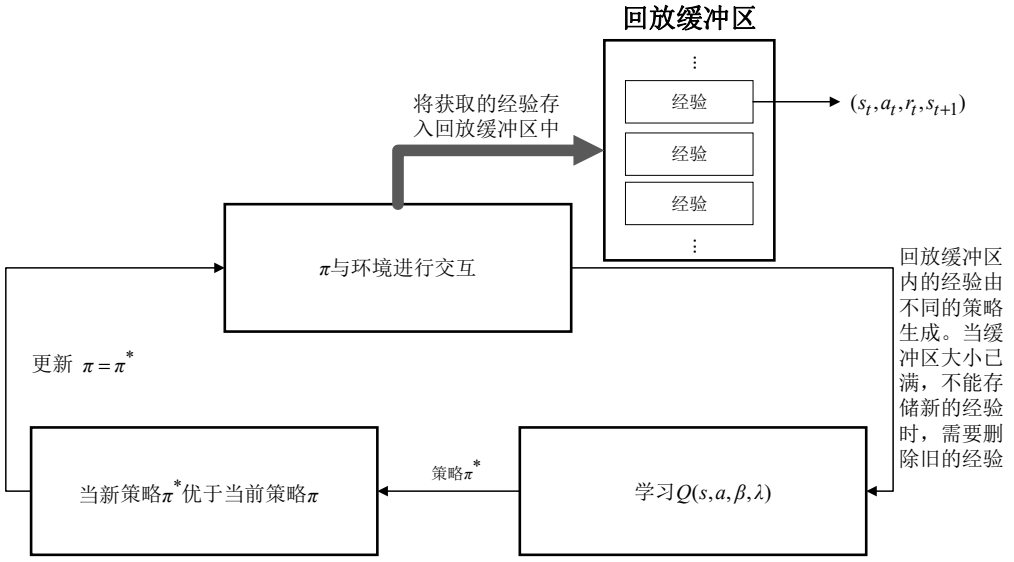


图 2-8 经验回放机制

在 Dueling DQN 算法中主要有ε-贪婪[58]和玻尔兹曼[59]两种常用的探索机制。玻尔兹曼探索机制是输出动作空间上的概率分布，根据概率分布采样选择动作，如公式(2-2)所示：

$$P(a|s)=\frac{\exp\left(Q(s,a,\beta,\lambda)/T\right)}{\sum_{a'}\exp(Q(s,a',\beta,\lambda)'/T)} \tag{2-2}$$

其中，$P(a|s)$表示概率分布的值；$T>0$表示温度系数，随着T的值不断减小，智能体采取的动作逐渐由探索转为利用操作。本文使用的是ε-贪婪探索机制如公式(2-3)所示：

$$a_t=\begin{cases} argmax_a Q_{policy}\left(\Phi(s_t),a,\theta\right), & if\ x<1-\varepsilon \\ random.random(), & otherwise \end{cases}. \tag{2-3}$$

其中，x表示[0,1]之间的随机变量，当$x<1-\varepsilon$时，智能体执行利用动作，否则执行探索动作；$\varepsilon\in[0,1]$表示可调参数，模型训练开始时ε设置为 1，随着训练步长的增加，ε以线性方式逐渐减小到e_{min}，如公式(2-4)所示：

$$\varepsilon=e_{max}+u\bullet\left(e_{min}-e_{max}\right) \tag{2-4}$$

其中，$u=min(1,\frac{steps}{totalSteps})$表示利用因子；$steps$表示当前训练步长；$totalSteps$表示衰减$\varepsilon$-贪婪探测机制总步长，随着训练步长$steps$的增大，$u$将以线性方式逐渐增加到 1；$e_{max}$和$e_{min}$分别表示表示最大和最小探索因子。

§2.2.2 DDPG 算法

DDPG是一种基于策略的DRL算法，被广泛用于解决连续动作空间的决策问题。

DDPG 算法可以简单理解成是 DQN 算法加上演员-评论家（AC，Actor-Critic）的一种框架算法。如图 2-9 所示，DDPG 借鉴了 DQN 经验回放机制的优势，用于经验样本的高效采样，减少了获取经验样本的时间开销，有效提高了网络模型的学习性能。为了解决 DQN 在更新 Q 网络时存在过估计的问题，DDPG 采用一种 AC 架构用于更新 Q 网络。具体来讲，Actor 中的策略网络根据$\pi_\theta(s_t)$选择当前状态s_t下的动作a_t,并与环境进行交互获取(s_t, r_t, s_{t+1})，通过公式(2-5)计算网络模型的损失值并更新策略网络的参数θ^P；策略网络的目标网络 Target_P 根据$\pi_{\theta'}(s_{t+1})$，从回放缓冲区中随机采样$batch$大小的经验样本用于选择s_{t+1}下的动作a_{t+1}，并在训练一定的步长后，根据公式(2-6)更新目标网络的参数$\theta^{P'}$。Critic 中的 Q 网络根据$Q_{\theta^Q}(s_t, a_t)$计算 Q 值，通过公式(2-7)计算网络模型的均方损失值并更新 Q 网络的参数θ^Q;Q 网络的目标网络 Target_Q 用于计算$Q_{\theta^{Q'}}(s_{t+1}, a_{t+1})$，并在训练达到一定的步长后根据公式(2-8)更新目标网络的参数$\theta^{Q'}$。其中，τ为更新系数，一般设置为接近于 1 的数值，用于防止全量复制以保证目标网络参数的更新幅度，进而提高算法模型的稳定性。

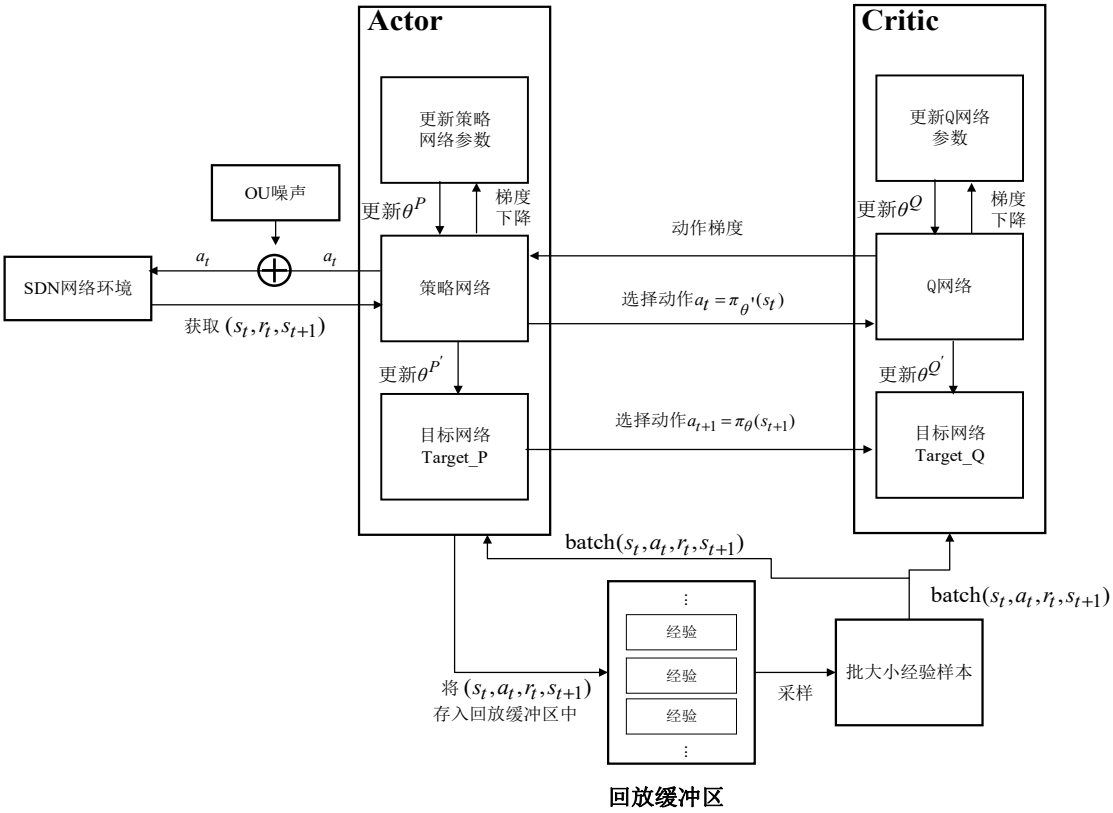


图 2-9 DDPG 算法结构图

$$loss1 = -Q_{\theta^P}\left(s_t, a_t\right) \tag{2-5}$$

$$\theta^{P'} = \tau \bullet \theta^P + (1-\tau) \bullet \theta^{P'} \tag{2-6}$$

$$loss2 = MSE\left[Q_{\theta^Q}\left(s_t, a_t\right), r_t + \gamma Q_{\theta^{Q'}}\left(s_{t+1}, a_{t+1}\right)\right] \tag{2-7}$$

$$\theta^{Q'} = \tau \bullet \theta^{Q} + (1-\tau) \bullet \theta^{Q'} \tag{2-8}$$

DDPG 算法能够利用 AC 架构对网络生成的动作进行评价的特点，使得 DDPG 算法不需要像 DQN 算法一样采取ε-贪婪等探索机制来进行动作的选取。加上 DDPG 算法是一种基于异策略（Off-policy）训练方式的确定性策略，如果智能体使用相同的策略来进行探索，将导致智能体不会尝试足够多的动作空间来学习更加有用的决策，从而影响网络模型的性能。因此，为了让 DDPG 算法中的智能体能够更好的进行探索学习，在训练的时候一般会对动作加上噪声，如本文所使用的 Ornstein-Uhlenbeck（OU）噪声，以提高智能体的感知能力和网络模型的性能。

§2.2.3 PPO 算法

PPO 算法是一种基于 AC 框架下的随机策略梯度算法，因此也是一种基于策略的 DRL 算法。PPO 算法是基于置信区域策略优化（TRPO，Trust Region Policy Optimization）[60]算法的改进，其核心思想是使用 Off-policy 方式来训练网络模型，每次更新网络参数时都需要与环境进行采样，确保策略朝着更好的方向进行优化，从而使得新策略比旧策略的累计期望收益更高，即每次更新都会使得智能体获取更大的奖励回报。

PPO 算法的训练过程如图 2-10 所示，Actor_New 网络根据策略$\pi = (s_t|\theta^a)$选择状态s_t下的动作，并与环境进行交互获取(r_t, s_{t+1})存放到回放缓冲区中。当缓冲区的经验样本满足$batch$大小时，通过公式(2-9)计算 Actor_New 网络的损失值并更新参数θ^a。其中，$\xi_t = \frac{\pi_{\theta^a}(a|s_t)}{\pi_{\theta^a_{old}}(a|s_t)}$表示新旧策略的比值，主要是用于解决 PPO 算法进行梯度下降时所面临的学习率设置大小问题，过大的学习率会减少模型的训练时长，但不能保证模型的性能，而过小的学习率能够保证模型的性能，但却增加了模型的训练时长。因此，通过设置比值的方式来限制新策略的更新幅度，从而使 PPO 算法在学习率较大时仍能保证较好的性能，同时减少模型的训练时长。A^{π}表示优势函数如公式(2-10)所示，其中$Q^{\pi}(s_t, a_t)$为状态-动作收益函数，如公式(2-11)所示，$V^{\pi}(s_t)$为状态收益函数。clip 表示裁剪函数用于控制ξ_t的输出值，如图 2-11 所示，当$\xi_t < 1-\sigma$时，输出$1-\sigma$，而当$\xi_t > 1+\sigma$时，输出$1+\sigma$（σ为常量，一般设置为 0.1 或 0.2）。当训练达到一定步长时，将 Actor_New 网络的参数更新到 Actor_Old 网络中。Critic 网络根据公式(2-12)计算出损失值并更新参数θ^c。

$$loss3 \approx \sum_{(s_t, a_t)} \min\{\xi_t \bullet A^{\pi}(s_t, a_t), \text{clip}(\xi_t, 1-\sigma, 1+\sigma) \bullet A^{\pi}(s_t, a_t)\} \tag{2-9}$$

$$A^{\pi}(s_t, a_t) = Q^{\pi}(s_t, a_t) - V^{\pi}(s_t) \tag{2-10}$$

$$Q^{\pi}(s_t, a_t) = r_t + \gamma \bullet V^{\pi}(s_{t+1}) \tag{2-11}$$

$$loss4 = r_t + \gamma \bullet V^{\pi}(s_{t+1}) - V^{\pi}(s_t) \tag{2-12}$$

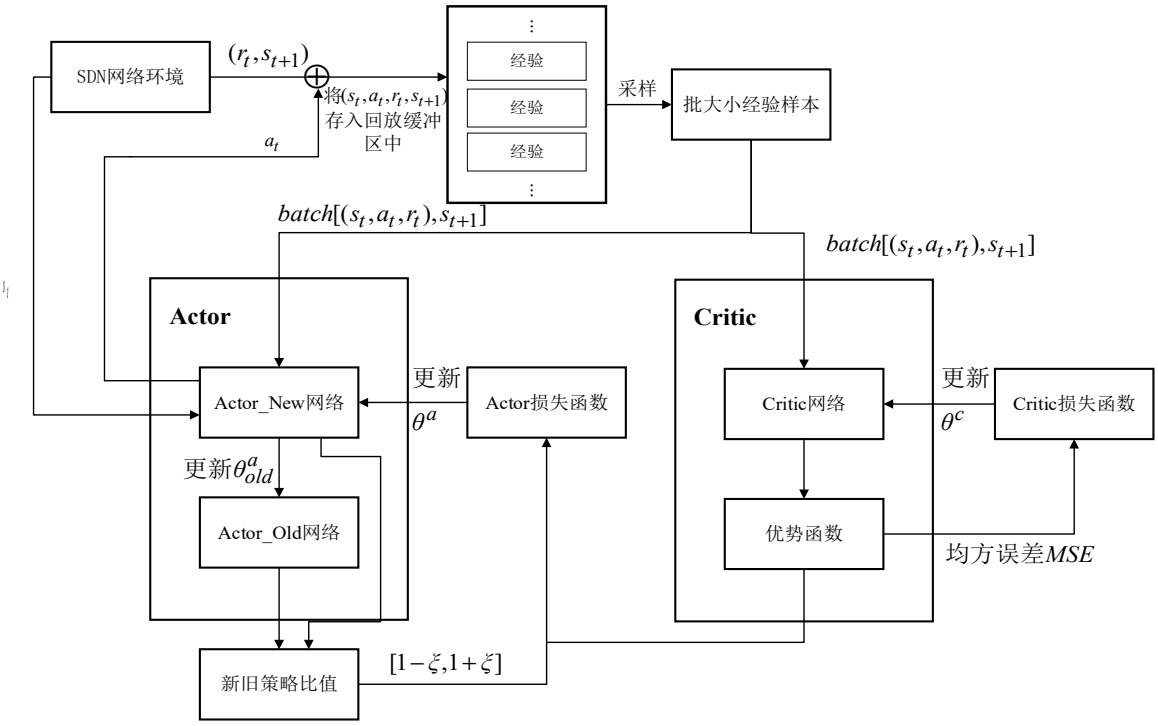


图 2-10 PPO 算法结构图

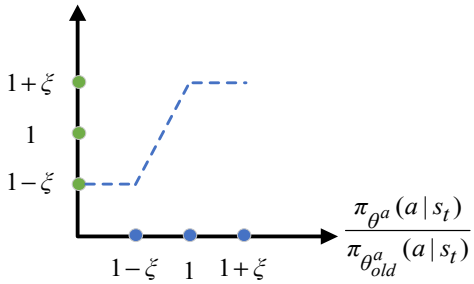


图 2-11 clip 裁剪函数

§ 2.3 网络流量状态预测

随着网络业务的发展，网络流量状态呈现出复杂、非线性的特点，基于线性网络流量状态预测模型已经很难适应当前网络流量状态的特点，导致对网络流量状态的预测效果并不理想，而基于神经网络的网络流量状态预测模型优势明显。传统神经网络是一种前馈网络结构，包括输入层、隐藏层和输出层，上一层神经元的输出作为下一层神经元的输入，这种传播机制只能让传统神经网络模型学习到当前时刻下的网络流量状态信息，而上一时刻的网络流量状态信息几乎不会影响网络模型的性能，从而导致传统神经网络模型不能有效解决时序化网络流量状态预测的问题。

与传统的神经网络不同，循环神经网络（RNN，Recurrent Neural Network）[61]使用一种内部反馈机制，能够较好解决时序化网络流量状态预测的问题，典型的 RNN

结构图如图 2-12 所示。其中，X表示输入层中的流量矩阵；h表示隐藏层中的矩阵；O表示输出层中的预测流量矩阵；W_X、W_h和W_O分别表示输入层、隐藏层和输出层的权重矩阵，作为 RNN 中每个神经元共享参数。在 RNN 模型中每层神经元的输入由当前层神经元的输入与前一层神经元的输出组成，即不仅与当前时刻下的状态信息有关，而且与上一时刻的历史状态信息相关，从而使得 RNN 具备记忆功能，有效解决了非线性时序化预测的难题，因此被广泛应用于网络流量状态的预测。隐藏层和输出层前向传播的计算公式，如公式(2-13)和(2-14)所示：

$$h_t = f\left(W_X \bullet X_t + W_h \bullet h_{t-1} + \omega_h\right) \tag{2-13}$$

$$O_t = g\left(W_O \bullet h_t + \omega_O\right) \tag{2-14}$$

其中，$f(\cdot)$和$g(\cdot)$表示激活函数；X_t表示当前时刻t下输入层中的流量矩阵；h_{t-1}和h_t分别表示当前时刻t和上一时刻$t-1$下隐藏层中的矩阵；O_t表示当前时刻t下输出层中的预测流量矩阵；ω_h和ω_O分别表示隐藏层和输出层中的偏置矩阵。

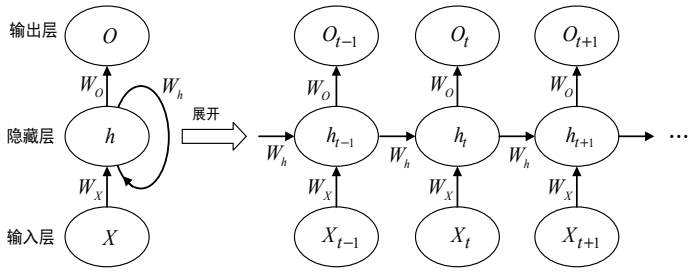


图 2-12 RNN 结构图

RNN 虽然在非线性时序化预测问题中表现较好，但随着当前与历史时刻的时序预测跨度不断增大，RNN 在进行反向传播求导的过程中，极易产生梯度消失和梯度爆炸的现象，从而影响 RNN 的时序化预测能力。因此，RNN 不适合处理具有长期依赖性的时序化预测问题。

§2.3.1 长短期记忆网络

RNN 凭借前向和反向传播的内部反馈机制，通过不断调整网络权重参数，能够很好的解决时序跨度较小的预测问题。然而，在实际生活中，预测问题变得越来越复杂，需要获取更大的时间跨度网络状态信息，才能保证网络预测模型性能的有效性。因此，亟需一种改进的 RNN 模型来解决上述问题。

长短期记忆网络（LSTM，Long Short-term Memory）[62]是一种改进的 RNN 模型，由遗忘门、输入门、输出门以及贯穿于整个时序跨度的细胞单元状态组成，如图 2-13 所示。LSTM 通过遗忘门决定前一个时序步长中哪些状态信息需要保留、输入门决定

当前时刻输入的状态信息那些是重要的和输出门决定下一个时序步长中神经元的隐藏状态，三种特殊门结构能够对网络预测模型中的神经元状态改变进行控制，有效解决了 RNN 在较大的时序跨度预测问题中出现梯度消失和梯度爆炸的现象。接下来，介绍 LSTM 中三种特殊门的主要功能。

遗忘门f_t主要用于决定丢弃或保留哪些状态信息，即对输入的时序化状态信息进行选择操作。如公式(2-15)所示，遗忘门f_t对上一时刻隐藏层中的状态信息和当前时刻输入层中的状态信息$[h_{t-1}, X_t]$，通过σ函数输出一个[0,1]之间的值来决定细胞状态C_{t-1}中信息保留程度。

$$f_t = \sigma\left(W_f \bullet \left[h_{t-1}, X_t\right] + \omega_f\right) \tag{2-15}$$

其中，σ为 sigmoid 函数；W_f和ω_f分别表示遗忘门中的权重矩阵和偏置矩阵。

输入门i_t主要用于确定当前时刻输入的状态信息中哪些是重要的。首先，将$[h_{t-1}, X_t]$通过σ函数输出一个[0,1]之间的值来决定i_t的保留程度，如公式(2-16)所示。然后，再将$[h_{t-1}, X_t]$通过$tanh$函数获取候选状态$\widetilde{C_t}$，如公式(2-17)所示。最后，通过公式(2-18)获取输入状态信息的重要程度C_t。

$$i_t = \sigma\left(W_i \bullet \left[h_{t-1}, X_t\right] + \omega_i\right) \tag{2-16}$$

$$\widetilde{C_t} = tanh\left(W_c \bullet \left[h_{t-1}, X_t\right] + \omega_c\right) \tag{2-17}$$

$$C_t = f_t \bullet C_{t-1} + i_t \bullet \widetilde{C_t} \tag{2-18}$$

输出门O_t主要用于确定下一个隐藏状态h_t。首先，将$[h_{t-1}, X_t]$通过σ函数输出一个[0,1]之间的值来决定O_t的保留程度，如公式(2-19)所示。然后，根据公式(2-20)得到下一个隐藏状态h_t。

$$O_t = \sigma\left(W_O \bullet \left[h_{t-1}, X_t\right] + \omega_O\right) \tag{2-19}$$

$$h_t = O_t \bullet tanh\left(C_t\right) \tag{2-20}$$

最终，将获取到的C_t和h_t传递到下一个时序步长的神经元中，从而不断更新 LSTM 网络模型的权重参数，实现对网络流量状态的预测能力。

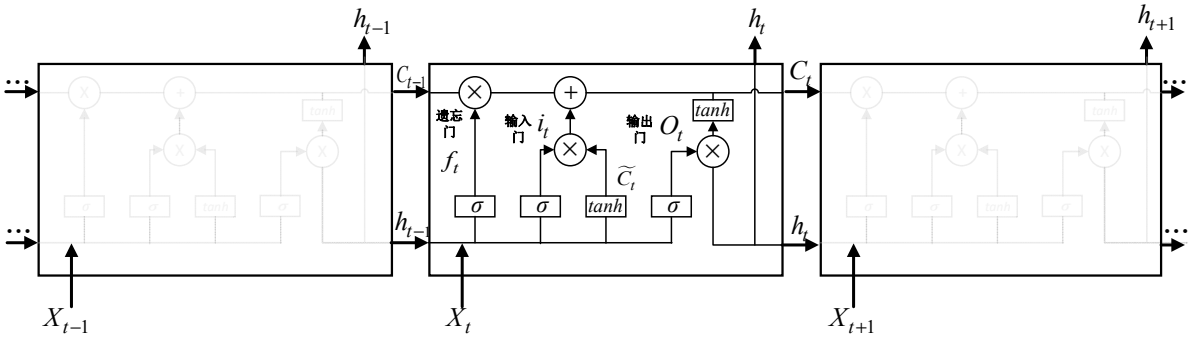


图 2-13 LSTM 结构图

§2.3.2 门控循环单元

门控循环单元（GRU，Gated Recurrent Unit）[63]是 LSTM 的一个变体。如图 2-14 所示，GRU 由更新门和重置门两个特殊门组成。相较于 LSTM 网络模型，GRU 网络模型中参数量更少，不仅能够减少预测模型过拟合的风险，而且能更快地使预测模型达到收敛状态，更好满足本文所设计的智能路由算法对于实时获取流量矩阵的需求。因此，本文采用 GRU 作为网络流量状态预测模型。接下来，介绍 GRU 中两种特殊门的主要功能。

更新门z_t可以简单理解成是 LSTM 中遗忘门和输入门两种特殊门的结合，它决定了丢弃或保留哪些状态信息以及状态信息的重要程度。如公式(2-21)所示，更新门z_t将上一时刻隐藏层中的状态信息和当前时刻输入层中的状态信息$[h_{t-1}, X_t]$，通过σ函数输出一个[0,1]之间的值来决定细胞状态z_t中信息的保留程度。

$$z_t = \sigma\left(W_z \bullet \left[h_{t-1}, X_t\right] + \omega_z\right) \tag{2-21}$$

重置门r_t的功能与更新门类似，用来确定下一个隐藏状态h_t。首先，根据公式(2-22)输出一个[0,1]之间的值来决定r_t保留程度。然后，将r_t与上一时刻h_{t-1}进行重置操作，获取候选隐藏状态，如公式(2-23)所示。其中，符号⊙表示将矩阵中向量对应的数值进行相乘后再相加。最后，根据公式(2-24)获取下一个隐藏状态h_t，将其传递到下一个时序步长的神经元中，不断更新 GRU 网络模型的权重参数，从而实现对网络流量状态的预测能力。

$$r_t = \sigma\left(W_r \bullet \left[h_{t-1}, X_t\right] + \omega_r\right) \tag{2-22}$$

$$\widetilde{h}_t = \tanh\left(W_{\tilde{h}} \bullet \left[r_t \odot h_{t-1}, X_t\right] + \omega_{\tilde{h}}\right) \tag{2-23}$$

$$h_t = \left(1 - z_t\right) \odot h_{t-1} + z_t \odot \widetilde{h}_t \tag{2-24}$$

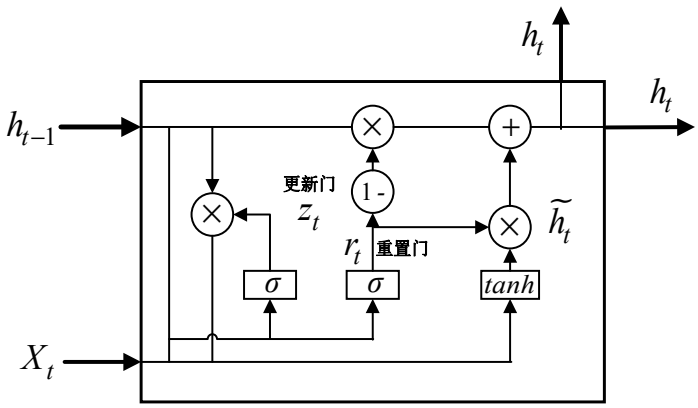


图 2-14 GRU 结构图

§2.4 网络性能评价指标

为了更好评估智能路由优化算法对网络性能的影响，本文设计了网络吞吐量、网络时延和网络丢包率三个性能评价指标。考虑到 SDN 网络长期运行可能产生一些波动，导致单次获取的网络吞吐量、网络时延和网络丢包率三个性能指标，不能准确反映当前时刻下的网络性能。因此，为了降低网络波动带来的影响，本文采用取平均值的方法来表示网络的性能指标。为了表述方便，本文使用网络吞吐量、网络时延和网络丢包率分别指代网络平均吞吐量、网络平均时延和网络平均丢包率。

§2.4.1 网络吞吐量

对于 SDN 单控制器管理下的网络链路吞吐量和网络吞吐量的计算公式分别如(2-25)和(2-26)所示：

$$throughput_{e_{ij}} = \frac{tx\_bytes\left(e_{ij}\right)}{bw_t\left(e_{ij}\right)\bullet\left|\Delta t\right|} \tag{2-25}$$

$$avg\_throughput = \frac{\sum_i^{|V|}\sum_j^{|V|} throughput_{e_{ij}}}{|V|\bullet|V|},\quad i,j=1,2,\ldots,|V| \tag{2-26}$$

其中，i和j表示 SDN 单控制器管理下的网络交换机节点名称；$tx_bytes(e_{ij})$表示在时间间隔Δt内交换机节点i发送到交换机节点j的字节数；$bw_t(e_{ij})$表示在Δt内链路e_{ij}的剩余带宽；$throughput_{e_{ij}}$表示在Δt内链路e_{ij}的吞吐量；$avg_throughput$表示 SDN 单控制器管理下的网络吞吐量。

对于 SDN 多控制器管理下的网络，存在多个网络子域。因此，本文为了更好的验证所提出的跨域智能路由算法的有效性，分别对每个网络子域中的域内网络指标和全局网络中的网络指标进行实验对比。每个网络子域中的域内网络吞吐量和全局网络吞吐量的计算公式分别如(2-27)和(2-28)所示：

$$domain\_avg\_throughput_{D_d} = \frac{\sum_i^{|V_{Dd}|}\sum_j^{|V_{Dd}|} throughtput_{D_{d_{e_{ij}}}}}{\left|V_{D_d}\right|\bullet\left|V_{D_d}\right|},\quad i,j=1,2,\ldots,\left|V_{D_d}\right| \tag{2-27}$$

$$all\_avg\_throughput = \frac{\sum_i^{\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|}\sum_j^{\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|} throughtput_{D_{1_{e_{ij}}}\cup D_{2_{e_{ij}}}\cup D_{3_{e_{ij}}}}}{\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|\bullet\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|},$$

$$i,j=1,2,\ldots,\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right| \tag{2-28}$$

其中，i和j表示 SDN 多控制器管理下的网络交换机节点名称；D_d $(d=1,2,3)$表示三

个网络子域; $throughtput_{D_d e_{ij}}$表示在时间间隔Δt内网络子域D_d $(d=1,2,3)$中的链路e_{ij}吞吐量；$throughtput_{D_1 e_{ij} \cup D_2 e_{ij} \cup D_3 e_{ij}}$表示全局网络中链路$e_{ij}$的吞吐量；$domain_avg_throughput_{D_d}$表示网络子域$D_d$ $(d=1,2,3)$的域内网络吞吐量；$all_avg_throughput$表示全局网络吞吐量。

§2.4.2 网络时延

对于 SDN 单控制器管理下的网络链路时延可以使用 SDN 网络时延的测量方式进行获取[64]。如图 2-15 所示，SDN 控制器使用链路发现协议（LLDP，Link Layer Discovery Protocol）分别获取时间$T1$和$T2$。然后，通过向交换机 S1 和 S2 周期性发送 Echo 报文，分别获取时间Ta和Tb。在获得时间$T1$、$T2$、Ta和Tb后，根据公式(2-29)计算出网络链路时延，进而获取 SDN 单控制器管理下的网络时延如公式(2-30)所示。

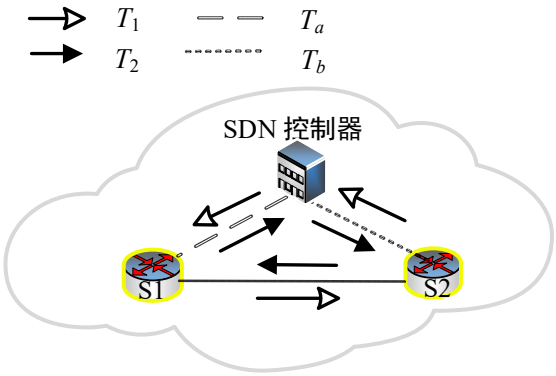


图 2-15 网络链路时延测量方式

$$delay_{e_{ij}} = \frac{(T_1 + T_2) - (T_a + T_b)}{2} \tag{2-29}$$

$$avg\_delay = \frac{\sum_i^{|V|} \sum_j^{|V|} delay_{e_{ij}}}{|V| \bullet |V|}, \quad i, j = 1, 2, \ldots, |V| \tag{2-30}$$

由于 SDN 多控制器管理下的网络存在多个网络子域，因此，需要分别获取每个网络子域中的域内链路时延和网络子域间的域间链路时延，才能获取 SDN 多控制器下的全局网络链路时延。每个网络子域中的域内链路时延按照如图 2-15 所示的方式进行获取，而网络子域间的域间链路时延按照如图 2-16 所示的方式进行获取。控制器 C1 管理下的交换机 S3 通过发送探测包$detect_pkt$给控制器 C3 管理下的交换机 S6 获得时间T_{inter1}，再由 S6 发送$detect_pkt$给 S3 获取时间T_{inter2}，然后根据公式(2-31)计算出域间网络链路时延，最后根据公式(2-32)获取全局网络链路时延。

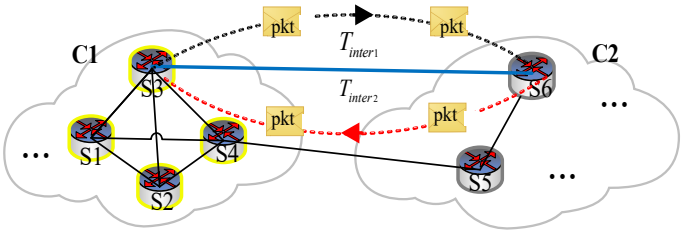


图 2-16 域间网络链路时延测量方式

$$delay_{e_{ij}} = \frac{T_{inter1} + T_{inter2}}{2} \tag{2-31}$$

$$delay_{D_{1_{e_{ij}}} \cup D_{2_{e_{ij}}} \cup D_{3_{e_{ij}}}} = \begin{cases} delay_{D_{d_{e_{ij}}}} & ,if\ e_{ij} \in D_d, d = 1,2,3 \\ delay_{e_{ij}} & ,otherwise \end{cases} \tag{2-32}$$

其中，$delay_{D_{de_{ij}}}$表示网络子域D_d $(d = 1,2,3)$的域内链路时延；$delay_{D_{1_{e_{ij}}} \cup D_{2_{e_{ij}}} \cup D_{3_{e_{ij}}}}$表示全局网络链路时延，当链路$e_{ij}$属于域内链路时使用$delay_{D_{de_{ij}}}$表示全局网络链路时延，而当链路$e_{ij}$属于域间链路时使用$delay_{e_{ij}}$表示全局网络链路时延。SDN 多控制器管理下的每个网络子域中的域内网络时延和全局网络时延的计算公式分别如(2-33)和(2-34)所示。

$$domain\_avg\_delay_{D_d} = \frac{\sum_i^{|V_{D_d}|} \sum_j^{|V_{D_d}|} delay_{D_{d_{e_{ij}}}}}{|V_{D_d}| \bullet |V_{D_d}|} \quad ,i,j = 1,2,\ldots,|V_{D_d}| \tag{2-33}$$

$$all\_avg\_delay = \frac{\sum_i^{|V_{D_1} \cup V_{D_2} \cup V_{D_3}|} \sum_j^{|V_{D_1} \cup V_{D_2} \cup V_{D_3}|} delay_{D_{1_{e_{ij}}} \cup D_{2_{e_{ij}}} \cup D_{3_{e_{ij}}}}}{|V_{D_1} \cup V_{D_2} \cup V_{D_3}| \bullet |V_{D_1} \cup V_{D_2} \cup V_{D_3}|} ,$$

$$i,j = 1,2,\ldots,|V_{D_1} \cup V_{D_2} \cup V_{D_3}| \tag{2-34}$$

§2.4.3 网络丢包率

基于 SDN 单控制器管理下的网络链路丢包率和网络丢包率的计算公式分别如(2-35)和(2-36)所示：

$$loss_{e_{ij}} = \frac{tx\_pkts(e_{ij}) - rx\_pkts(e_{ij})}{tx\_pkts(e_{ij})} \tag{2-35}$$

$$avg\_loss = \frac{\sum_i^{|V|} \sum_j^{|V|} loss_{e_{ij}}}{|V| \bullet |V|} \quad ,i,j = 1,2,\ldots,|V| \tag{2-36}$$

其中，i和j表示 SDN 单控制器管理下的网络交换机节点名称；$tx_pkts(e_{ij})$表示在时间间隔Δt内交换机节点i发送到交换机节点j的数据包数量；$rx_pkts(e_{ij})$表示在Δt内交

换机节点i接收到交换机节点j的数据包数量；$loss_{e_{ij}}$表示在Δt内链路e_{ij}的丢包率；$avg_throughput$表示网络丢包率。

对于 SDN 多控制器管理下的每个网络子域中的域内网络丢包率和全局网络丢包率的计算公式分别如(2-37)和(2-38)所示：

$$domain\_avg\_loss_{D_d}=\frac{\sum_i^{|V_{D_d}|}\sum_j^{|V_{D_d}|} loss_{D_{d_{e_{ij}}}}}{\left|V_{D_d}\right|\bullet\left|V_{D_d}\right|} \quad ,i,j=1,2,\ldots,\left|V_{D_d}\right| \tag{2-37}$$

$$all\_avg\_loss=\frac{\sum_i^{|V_{D_1}\cup V_{D_2}\cup V_{D_3}|}\sum_j^{|V_{D_1}\cup V_{D_2}\cup V_{D_3}|} loss_{D_{1_{e_{ij}}}\cup D_{2_{e_{ij}}}\cup D_{3_{e_{ij}}}}}{\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|\bullet\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right|} \quad ,$$

$$i,j=1,2,\ldots,\left|V_{D_1}\cup V_{D_2}\cup V_{D_3}\right| \tag{2-38}$$

其中，i和j表示 SDN 多控制器管理下的网络交换机节点名称；D_d $(d=1,2,3)$表示三个网络子域;$loss_{D_{d_{e_{ij}}}}$表示时间间隔Δt内网络子域D_d $(d=1,2,3)$中的链路e_{ij}丢包率；$loss_{D_{1_{e_{ij}}}\cup D_{2_{e_{ij}}}\cup D_{3_{e_{ij}}}}$表示全局网络中链路$e_{ij}$的丢包率；$domain_avg_throughput_{D_d}$表示网络子域$D_d$ $(d=1,2,3)$的域内网络丢包率；$all_avg_throughput$表示全局网络丢包率。

§2.5 本章小结

本章对本文涉及到的相关技术和理论进行了介绍说明。主要介绍了 SDN 架构的组成、OpenFlow 协议等 SDN 相关技术，同时还介绍了 Dueling DQN、DDPG 和 PPO 三种深度强化学习算法，并对 LSTM 和 GRU 两种网络流量状态预测算法进行了对比介绍。此外，还详细介绍了本文基于 SDN 单控制器和 SDN 多控制器管理下的网络吞吐量、网络时延和网络丢包率三种评价指标的设计思路。

第三章 基于 Dueling DQN 和网络流量状态预测的智能路由方法

本章针对传统路由方法仅使用有限网络链路信息进行路由决策，且适应动态复杂网络变化能力差、调整路由转发策略不灵活的缺陷，提出了一种基于 Dueling DQN 强化学习和网络流量状态预测的 SDN 智能路由方法（DRL-TP，**D**eep **R**einforcement **L**earning-network **T**raffic state **P**rediction），通过获取全局网络链路状态信息，实现实时自适应生成最佳路由转发决策。本章内容分为五小节：3.1 小节给出本章的问题描述；3.2 小节对本章 SDN 智能路由优化架构与建模进行了介绍；3.3 小节介绍了本章智能路由算法的详解设计过程；3.4 小节给出本章实验结果与实验分析；3.5 小节总结了对本章的研究工作。

§ 3.1 问题描述

随着各种新型网络设备不断出现，网络流量呈现指数型增长趋势。Cisco 公司在《思科视觉网络索引》中指出[65]，2023 年全球网络的流量总量已超过 4.8ZB，面对如此高量的网络流量，如何保证网络路由的实时自适应转发，是保证网络性能和提升用户体验的关键性因素。传统 TCP/IP 网络是一种分布式架构，由于控制和转发紧耦合、分散式管理的特点，加上不同厂商考虑其设备的安全性而存在不同的网络规则，导致难以获取网络的全局状态信息，仅使用有限的网络状态信息进行路由转发操作，难以保证网络的性能。SDN 作为一种新型网络架构，一方面通过将数据平面与控制平面分离，并使用 SDN 控制器进行集中统一管理，能够很方便的获取全局网络状态信息，另一方面通过网络开放可编程的优势，十分有利于智能化路由决策的部署与实现。

人工智能技术的兴起，使得 DRL 算法在解决复杂动态的网络路由优化问题中展示出巨大的优势，不少研究学者已经在 SDN 环境中部署 DRL 算法，用于实现自适应智能化路由转发决策，有效提升了网络的性能。然而，现有的方法仅考虑带宽等单个网络链路指标用于构建流量矩阵作为深度强化学习中状态空间设计，并且未考虑网络流量状态的未来变化趋势，这些都会对 SDN 智能路由优化算法的性能造成一定影响。因此，本章综合考虑带宽、时延、丢包率等六个网络链路指标用于构建流量矩阵，并使用 GRU 网络流量状态预测算法用于发现隐藏和未知的流量矩阵以提升智能路由算法的感知能力，从而实现实时自适应智能化的路由决策，有效提升了网络的性能。

§ 3.2 SDN 智能路由优化架构与建模

§ 3.2.1 系统整体架构

本章设计的 SDN 网络架构下智能路由方法模型的整体结构如图 3-1 所示，主要包括数据平面、控制平面、管理平面和知识平面，下面将逐一介绍所设计四个平面的功能。

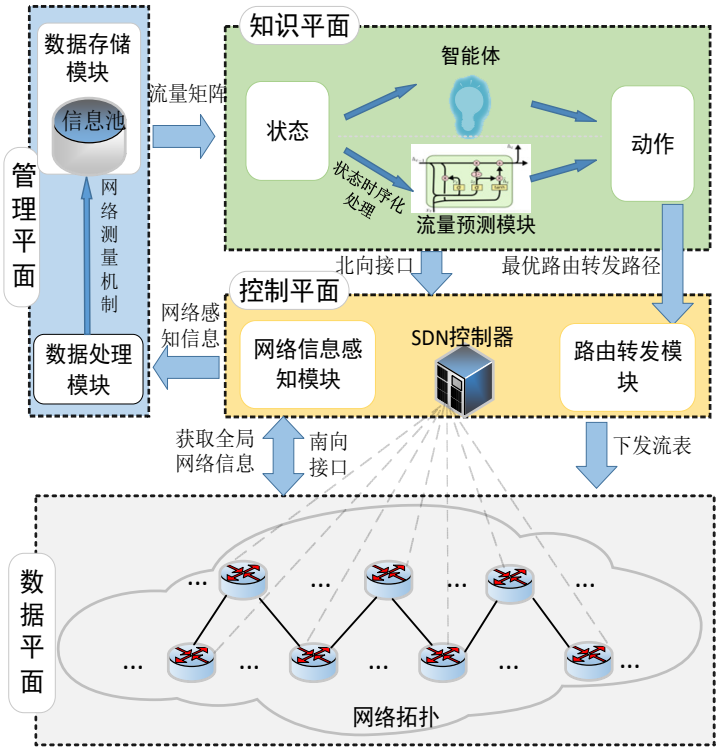


图 3-1 智能路由方法整体架构

§ 3.2.2 数据平面

在 SDN 网络架构中，数据平面主要由各种基于 OpenFlow 协议的转发设备构成。数据平面是执行网络数据包处理的实体，它只负责根据控制平面下发的指令对相应的数据包进行转发或丢弃操作，数据平面的可编程能力对 SDN 网络的部署和维护带来了极大便利。在本章设计的 SDN 智能路由方法中，数据平面的主要功能为：（1）通过响应控制平面周期性发送的相应请求提供网络全局的感知信息，例如：交换机端口流速、发送及接收字节数等;（2）智能体生成最优路由转发路径，经控制平面生成最优路由转发策略后，数据平面根据最优路由转发策略对数据包进行转发操作。

§3.2.3 控制平面

控制平面是 SDN 架构的中枢，它通过南向接口与数据平面交互，周期性的发送相应的请求指令以获取全局网络拓扑和链路信息，通过北向接口处理上层应用的请求，并为上层应用提供服务。在本章 SDN 智能路由方法模型中，控制平面主要部署了两个模块：网络信息感知模块和路由转发模块。网络信息感知模块的主要功能为：（1）通过 LLDP 协议获取全局网络中的拓扑信息；（2）通过周期性发送相应的 Request 请求指令，获取网络中各交换机的端口状态信息并构成网络感知信息。路由转发模块主要功能为：（1）根据知识平面生成的最优路由转发路径，按照已获取的网络拓扑找到对应的主机节点；（2）根据最优转发路径和对应的主机节点，生成最优路由转发策略并下发到数据平面。

§3.2.4 管理平面

在本章 SDN 智能路由方法模型中，为 SDN 架构添加了一个管理平面，主要包含一个数据处理模块和数据存储模块。管理平面在获得控制平面的感知信息后，通过数据处理模块将其转换成带宽、时延、丢包率、已用带宽等网络链路状态信息并保存到数据存储模块的信息池中。管理平面包含了大量数据监测计算、转化存储等处理操作，是确保本章方法性能和网络稳定运行的关键。因此，在管理平面上部署了一个基于 SDN 多线程网络测量机制如图 3-2 所示。该测量机制上使用了多线程技术，一方面是由于在 SDN 环境中网络状态测量机制与控制器下发流表操作处于同一线程时，会导致网络不能长期正确运行，而且知识平面需要根据当前网络状态实时做出最优路由转发策略，这就需要与管理平面进行频繁的交互以获取当前时刻下的网络状态信息，而基于单线程的测量机制很难满足这种交互；另一方面数据处理和信息存储操作分别由不同线程执行，不仅能够准确及时处理和存储网络链路信息，而且能够在算法模型线下训练时，充分利用处理器等硬件资源。通过后续实验结果可以证明，基于多线程的网络测量机制，不仅能够有效获取网络链路状态信息，而且能够提升智能路由方法的性能。

SDN 多线程网络测量机制的主要流程为：创建信息探测线程，该线程每隔t秒就会向控制平面请求监测当前时刻的全局网络感知信息，控制平面收到监测请求后，通过发送相关的 Request 请求指令将获取到的全局网络感知信息返回。然后将获取的网络感知信息传入数据处理线程，通过相关等式转换成带宽、时延、丢包率等网络链路信息用于构建当前时刻的流量矩阵。数据处理线程是 SDN 网络测量机制的核心，它根据智能路由方法中训练测试和应用部署两种模式分别执行两种不同的职能。如图 3-

2 蓝色箭头序号所示，在训练测试模式中，将每一时刻构成的流量矩阵，用信息存储线程保存到数据存储模块的信息池中，供知识平面进行模型线下训练使用。如图 3-2 红色虚线序号所示，在应用部署模式下，将当前时刻的流量矩阵传入知识平面中，根据训练好的模型生成最优路由转发路径，经控制平面生成最优路由转发策略并进行下发流表操作，然后使用 SDN 网络测量机制获取当前网络状态下的链路信息，最后根据相关等式计算得到网络吞吐量、时延、丢包率三个评价指标，并用信息存储线程保存到存储模块的信息池中，供后本章续进行对比实验使用。

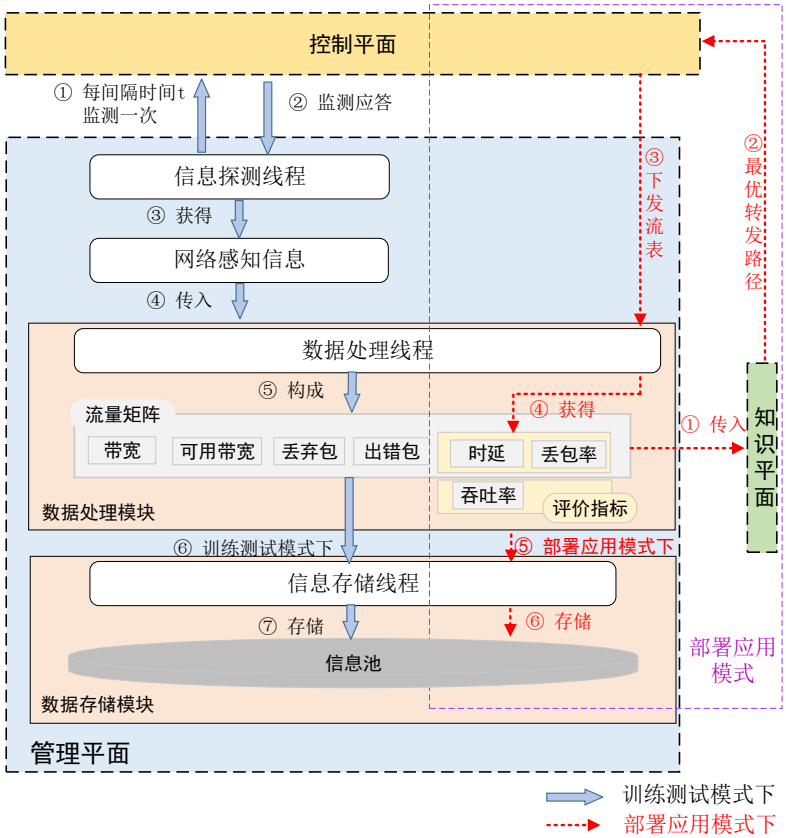


图 3-2 SDN 多线程网络测量机制

§3.2.5 知识平面

在本章 SDN 智能路由方法中，为 SDN 架构添加了一个知识平面。将知识平面（KP，Knowledge Plane）添加到 SDN 体系架构中，最早由 Clark 等人[66]提出，随后 Mestres 等人[67]根据 Clark 等人在 SDN 体系结构背景下定义的知识平面概念，提出了知识定义网络（KDN，Knowledge Define Network）的概念，指出知识平面能够将决策行为和推理过程集成到 SDN 网络架构中，提供描述、学习和智能化等功能，从而支持路由决策过程。从根本上讲，知识平面就是将管理平面上获取的全局网络链路状态信息，通过强化学习把学习到的网络动作行为转化为知识，然后根据这些知识智能

化的为网络做出相应策略。在本章所提出的智能路由方法模型中，知识平面主要包括深度强化学习模块和网络流量状态预测模块。深度强化学习模块，即智能体，主要功能是搭建 DRL 环境，将管理平面获取的流量矩阵转化成状态空间，利用基于端到端、无模型的深度强化学习，与网络环境进行不断交互，让智能体朝着更高奖励值的方向学习。当算法模型训练趋于收敛后，给出当前时刻下网络状态的动作，即为最优路由转发路径。

知识平面要实时获取流量矩阵以做出路由决策，这就需要 SDN 控制器频繁、不间断响应处理网络中的数据流。然而，随着应用业务类型的增多和用户需求的增加，网络中将会存在大量不同类型的数据流，这样的处理机制无疑会给控制器带来巨大的负载，可能致使网络中一些数据流长时间处于队列等待状态，不能得到及时处理，甚至还会出现回路循环现象，导致网络出现波动，影响 SDN 网络的正确运行。因此，本章设计的 SDN 网络测量机制采用多线程间隔式的方式获取网络全局链路状态，不仅有效解决了控制器连续处理大量、不同类型数据流带来的高负载问题，而且还能较好的满足知识平面实时获取流量矩阵的需求，但还是会存在对网络中一些流量矩阵监测的遗漏，从而影响智能路由方法的性能。为了解决间隔式 SDN 多线程网络测量机制对网络中一些流量矩阵存在监测遗漏的问题，本章在知识平面中添加了一个网络流量状态预测模块，通过该模块一方面能够预测出监测遗漏的流量矩阵，另一方面能够发现网络中隐藏和未知的网络流量状态。

§ 3.3 DRL-TP 智能路由算法设计

本章数据平面中的网络拓扑是一个无向图，可以用$G = \{V，E，W\}$表示，其中，V代表网络拓扑的 SDN 交换机节点，E代表网络拓扑中交换机节点之间的链路，W代表链路的权重大小，一般设置为相同的常量。DRL-TP 智能路由算法总体流程如图 3-3 所示。主要由 DRL 算法与网络流量状态预测算法组成，首先，通过 SDN 多线程网络测量机制获取流量矩阵，由网络流量状态预测算法获取预测流量矩阵，组成 DRL 算法的状态空间；然后，分别对深度强化学习算法和网络流量状态预测算法进行线下训练，将训练好的两个算法模型组成 DRL-TP 算法模型；最后，根据发送流大小实时获取当前网络状态下的流量矩阵，经过 DRL-TP 智能路由算法生成最优路由转发路径。接下来分别对设计的深度强化学习算法、网络流量状态预测算法、DRL-TP 智能路由算法进行介绍。

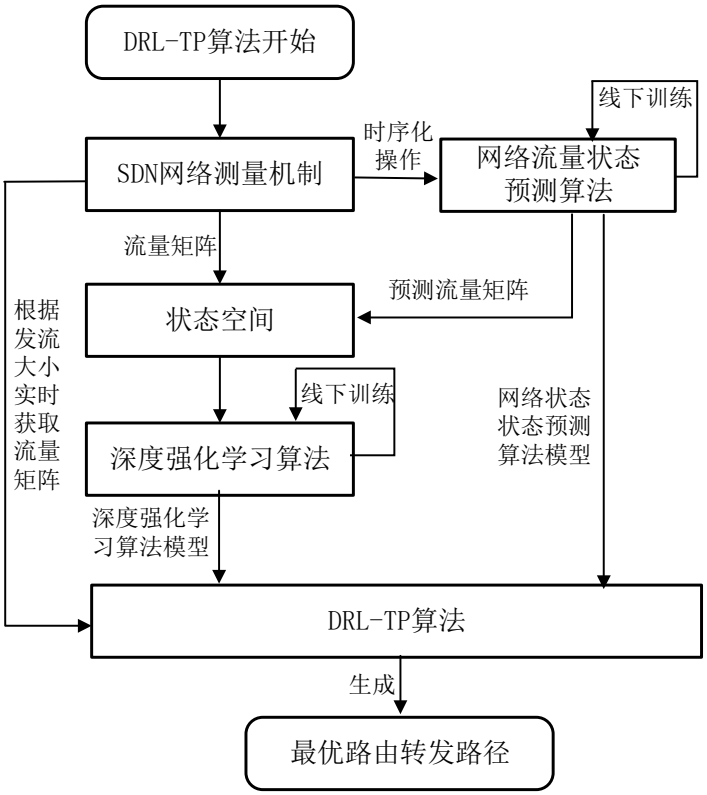


图 3-3 DRL-TP 智能路由算法总体流程

§3.3.1 深度强化学习算法

深度强化学习算法是一种框架性的算法，对于不同问题、不同应用领域需要设计不同的状态空间、动作空间和奖励函数。下面介绍本章基于深度强化学习框架下的 DRL-TP 智能路由算法中状态空间、动作空间和奖励函数的设计。

状态空间（S）。状态空间可表示为$S = TM$，其中TM表示在间隔t内的流量矩阵，由多个二维矩阵$M_{|V|\times|V|}$拼接而成，如公式(3-1)所示：

$$m_{ij} = w_1 \bullet \frac{1}{L_{bw_{ij}}} + w_2 \bullet L_{delay_{ij}} + w_3 \bullet L_{loss_{ij}} + w_4 \bullet L_{used_{bwij}} + w_5 \bullet L_{drops_{ij}} + w_6 \bullet L_{errors_{ij}} \quad ,i,j = 1,2,\ldots,|V| \tag{3-1}$$

其中，m_{ij}表示流量矩阵中的元素，由L_{bw}、L_{delay}、L_{loss}、L_{used_bw}、L_{drops}和L_{errors} 6 个方面的网络链路剩余带宽、时延、丢包率、已用带宽、弃包数和错包数信息矩阵元素通过相加映射构成，每个网络链路信息矩阵包含了当前时刻所有交换机节点之间的链路信息；$w_l \in [0,1], l = 1,2,..,6$表示可调参数，为构成流量矩阵元素的权重因子；$i$和$j$表示网络拓扑中交换机节点名称；$|V|$表示网络拓扑中交换机节点数量。流量矩阵结构图如图 3-4 所示。

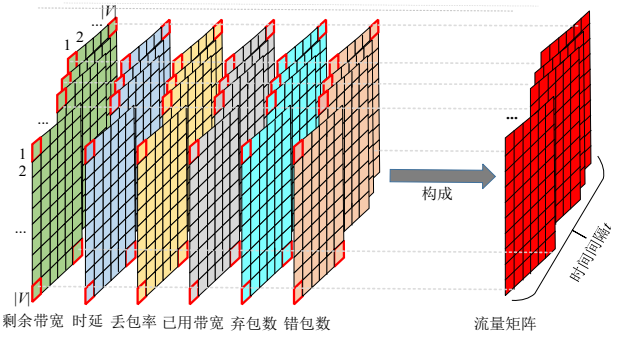


图 3-4 流量矩阵结构图

考虑到每个网络链路信息矩阵中的元素值存在较大差异，不能客观反映出各个网络链路信息矩阵对所组成流量矩阵的贡献值，甚至通过相加映射后组成的流量矩阵可能会受到某个较大链路信息矩阵的干扰，造成 DRL-TP 智能路由算法训练过程出现过多的波动难以收敛，因此需要进行对流量矩阵中的元素进行归一化的操作。对此，使用 Min-Max[68]技术对流量矩阵进行归一化操作，将矩阵中的元素归一化到指定区间$[a,b]$范围内，具体设计形式如公式(3-2)所示：

$$\overline{m_{ij}} = a + \frac{\left(m_{ij} - \min\left(TM\right)\right)\bullet\left(b-a\right)}{\max\left(TM\right) - \min\left(TM\right)} \tag{3-2}$$

其中，$\overline{m_{ij}}$表示流量矩阵归一化后的元素；$min(TM)$和$max(TM)$分别表示流量矩阵中最小和最大的元素。

动作空间（A）。动作空间中的动作主要以转发链路权重值和转发路径两种方式构成，前者不需要存储庞大的动作空间，但需要使用相关方法进一步转换成转发路径；后者直接输出转发路径，但需要存储庞大的动作空间，一种有效的解决方法是选择候选路径集作为动作空间，文献[26][31][33][43]已经证明了这种方法的有效性。本章设计的动作空间是采用后者直接输出转发路径的方式上改进而得。每个动作$a_t \in [0,1,\ldots,k]$对应于$s_t \in S$状态下的转发路径选择，由$k = \lfloor 0.1 \cdot |V| \cdot |V| \rfloor$个候选路径矩阵$C_{|V|\times|V|}$组成，每个候选路径矩阵中包含了所有源交换机节点到目的交换机节点的路径。其中，每个候选路径矩阵中的元素为交换机节点i到交换机节点j的路径$path_{ij} = [i,\ldots,j]$。

奖励函数（R）。奖励是指智能体执行相应的动作后产生的收益指标，用于引导智能体往更高的奖励回报的方向进行学习。在 DRL-TP 智能路由算法模型中，奖励函数如公式(3-3)所示，优化的目标主要是最大化剩余带宽，最小化时延、丢包率、已用带宽、弃包数及错包数，而 DRL 是以不断获取最大奖励值为目标。因此，对于最大化指标赋予正相关系数 1，对于最小化指标赋予负相关系数-1，$\varphi_l \in [0,1], l = 1,2,..,6$表示可调参数，为构建奖励函数的权重因子。

$$R = \varphi_1 \bullet L_{bw} - \varphi_2 \bullet L_{delay} - \varphi_3 \bullet L_{loss} - \varphi_4 \bullet L_{used_{bw}} - \varphi_5 \bullet L_{drops} - \varphi_6 \bullet L_{errors} \tag{3-3}$$

本章使用 Dueling DQN 作为深度强化学习算法，具体实现细节如算法 3-1 所示，首先输入相关超参数。其中，流量矩阵TM包括 SDN 多线程网络测量机制得到的流量矩阵和网络流量状态预测模块生成的预测流量矩阵。步骤 1：初始化两个结构相同的策略网络Q_{policy}和目标网络Q_{target}以及用于存储经验样本的回放缓冲区M。步骤 5-步骤 13：主要进行 Dueling DQN 网络模型的更新操作，当经验池中的数据大小满足$batch$时，利用网络Q_{policy}和Q_{target}分别获取估计值p_value和目标值t_value，然后根据梯度下降和反向传播调整Q_{policy}的权重和偏差，并使用均方损失函数计算网络模型的损失值。步骤 14-步骤 16：在训练达到一定步长后，将Q_{policy}的权重和偏差更新到Q_{target}中。步骤 17：将智能体移至下一个状态。在每次训练结束时，都会获得当前网络状态中所有源-目的交换机的路由转发路径。

算法 3-1 Dueling DQN 深度强化学习算法

输入:

学习率: lr

采样大小: $batch$

折扣因子: γ

权重因子: $\varphi_l \in [0,1], l = 1,2,\dots,6$

衰减参数: ε

衰减率: $decay$

目标网络更新频率: $freq$

训练总数: $episodes$

流量矩阵: TM

输出:

网络中所有源-目的交换机节点的转发路径

1 初始化Q_{policy}及Q_{target}网络权重θ、经验池M

2 **For** $episodes \leftarrow$ 1 to n do:

3 　智能体获取网络环境初始化状态s_t

4 　**While** next state s_{t+1} is not final state do:

5 　　更新衰减参数$\varepsilon = \varepsilon - (steps \cdot decay)$

6 　　根据公式(2-3)智能体获取当前状态s_t下的动作a_t

7 　　根据公式(3-3)智能体获取当前奖励值$r_t \leftarrow R(s_t, a_t)$

8 　　将经验Experiences$_t = (s_t, a_t, r_t, s_{t+1})$存放到$M$

9 　　**If** len$(M) \geq batch$ then:

10 　　从M中随机采样大小数据

11 　　根据价值函数公式(2-1)分别获取p_value以及t_value

$$p\_value = Q_{policy}(s_t, a_t, \theta)$$

$$t\_value = \begin{cases} r_t & ,if\ next\ state\ is\ final\ state \\ r_t + \gamma \cdot max_{a'} Q_{target}(s_{t+1}, a'; \theta), otherwise \end{cases}$$

12 　　使用$loss = (t_value - p_value)^2$执行梯度下降更新$Q_{policy}$ 网络权重参数θ

13 　　**End if**

If $steps \,\%\, freq \;==\; 0$ then:
更新Q_{target}网络模型参数，$\theta^{Q_{target}} \leftarrow \tau \cdot \theta^{Q_{policy}} + (1-\tau) \cdot \theta^{Q_{target}}$
End if
$s_t \leftarrow s_{t+1}$
End while
End for

§3.3.2 网络流量状态预测算法

本章使用 GRU 作为网络流量状态预测模型。一方面它相较于 LSTM 参数量少，能够减少预测模型过拟合的风险；另一方面，它能更快地使预测模型达到收敛状态，更好满足本章设计的知识平面对于实时获取流量矩阵的需求。GRU 网络流量状态预测算法，如算法 3-2 所示。首先输入相关超参数，其中输入层、隐藏层和输出层维度是组成 GRU 算法模块的基本参数。步骤 2：将流量矩阵经过时序化操作后生成TM_{input}和TM_{target}，分别作为 GRU 算法模型的输入矩阵和目标矩阵。步骤 3-步骤 10：进行 GRU 算法模型的训练，使用梯度下降和反向传播算法更新 GRU 网络的权重和偏差，并使用均方损失函数计算网络模型的损失值。当训练完成后，使用训练好的 GRU 算法模型生成预测流量矩阵，作为算法 3-1 中流量矩阵TM的组成部分。

算法 3-2 网络流量状态预测算法

输入:
学习率: lr
采样大小: $batch$
输入层维度: $input\_dim$
隐藏层维度: $hidden\_dim$
输出层维度: $output\_dim$
时序周期: seq
目标网络更新频率: $freq$
训练总数: $episodes$
流量矩阵: TM

输出:
预测流量矩阵

初始化 GRU 网络权重θ
将流量矩阵进行时序化操作，得到两个时序化流量矩阵
时序化输入流量矩阵： $TM_{input} \leftarrow TM$
时序化目标流量矩阵： $TM_{target} \leftarrow TM$
For $episodes$ ←1 to n do:
初始化$hidden$
For t ←0 to len(TM_{input}) do:

6 $TM_{output}^{t+seq+1}, hidden_{output} \leftarrow GRU(TM_{output}^{t,t+seq}, hidden)$

7 使用$loss$=$(TM_{output}^{t+seq+1} - TM_{target}^{t+seq+1})^2$执行梯度下降更新 GRU 网络权重参数$\theta$

8 $hidden \leftarrow hidden_{output}$

9 **End for**

10 **End for**

§ 3.3.3 DRL-TP 智能路由算法

DRL-TP 智能路由算法主要由 Dueling DQN 深度强化学习算法和 GRU 网络流量状态预测算法组成，如算法 3-3 所示。首先，输入训练完成的 Dueling DQN 和 GRU 算法模型，其中bw_list包含了每次发送流的大小。步骤 1：加载训练完成的 Dueling DQN 和 GRU 算法模型并组成 DRL-TP 智能路由算法模型。步骤 2-步骤 5：根据当前发送流的大小，使用 SDN 多线程网络测量机制实时获取当前网络状态下的流量矩阵TM，然后根据 DRL-TP 算法生成当前网络状态下所有源-目的交换机节点的最优路由转发路径all_paths，由控制平面生成最优路由转发策略并下发流表，最后数据平面按照最优转发策略进行数据包转发操作。

算法 3-3 DRL-TP 智能体路由算法

输入:

Dueling DQN 算法模型

GRU 预测算法模型

发流大小: bw_list

输出:

所有源-目的交换机节点的最优路由转发路径

1 加载 Dueling DQN 及 GRU 预测算法模型，构建 DRL-TP 算法模型

2 **For** $bw \leftarrow$ 1 to bw_list do:

3 获取当前网络状态下的流量矩阵TM

4 $all_paths \leftarrow$DRL-TP(TM)

5 **End for**

§ 3.4 实验与结果分析

本节将对本章所提出 DRL-TP 智能路由优化算法进行系列实验对比并验证其有效性。

§3.4.1 实验环境与测试

本章在一台 Ubuntu16.04 内存大小为 2G，4 核处理器的系统下，安装了 Mininet 2.3.0[69]和 Ryu4.34[70]用于搭建 SDN 网络环境，其中 Mininet 用于搭建 SDN 网络拓扑，Ryu 作为 SDN 控制器，Iperf[71]用于模拟网络数据流的发送，如图 3-5 所示，采用均匀分布等概率的方式发送网络数据流大小，共获取了 1616 个流量矩阵，等概率函数如公式(3-4)所示：

$$prop(f)=\frac{1}{\eta_2-\eta_1},\quad \eta_1<f<\eta_2 \tag{3-4}$$

其中，η_1为 5，η_2为 100，f表示数据流大小，$prop(f)$表示选取当前发送数据流大小的等概率函数。实验所使用的网络拓扑是修改后的纽约市中心网络[72]，如图 3-6 所示，总共包含 14 个节点，每个节点为一台支持 OpenFlow1.3 协议的交换机，每台交换机挂载一台主机。为了满足新型网络需求和搭建异构网络环境，交换机之间的链路带宽大小在[15Mbit-100Mbit]范围内随机设置。

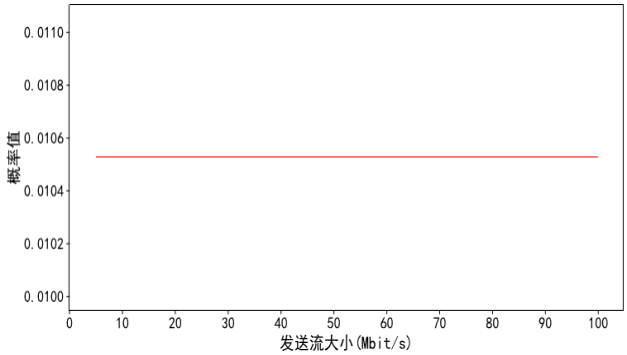


图 3-5 均匀分布等概率发送数据流大小

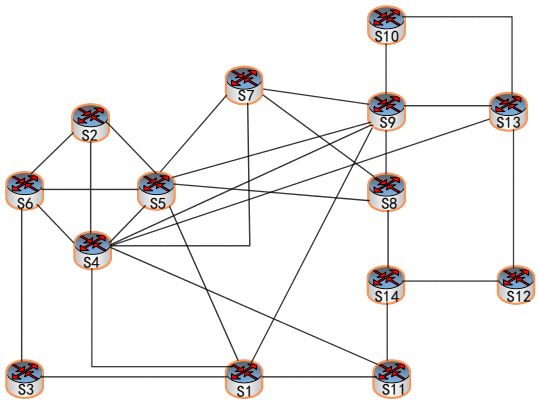


图 3-6 修改后 14 个节点的纽约市中心网络拓扑图

§3.4.2 预测算法性能与实验参数分析

首先分析 GRU 网络流量状态预测算法对 SDN 智能路由方法性能的影响。从图 3

-7 可以看出，智能体使用 GRU 预测算法获得的奖励值明显比不使用 GRU 预测算法高。这是由于 GRU 预测算法一方面能够发现 SDN 网络中隐藏和未知的网络流量状态，而基于 SDN 技术的网络测量机制一般很难获取这些隐藏和未知的网络流量状态，另一方面 GRU 预测算法能够预测未来一段时间内网络流量状态的变化趋势，因此使用 GRU 网络流量状态预测算法，能够让 DRL-TP 智能路由算法探索到更大的状态空间，学习到更优的行为动作，从而获得更高的奖励值，验证了使用 GRU 网络流量状态预测算法能够在一定程度上提升 DRL-TP 智能路由算法的性能。

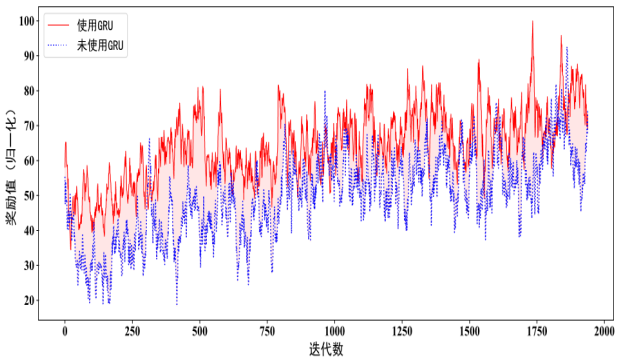
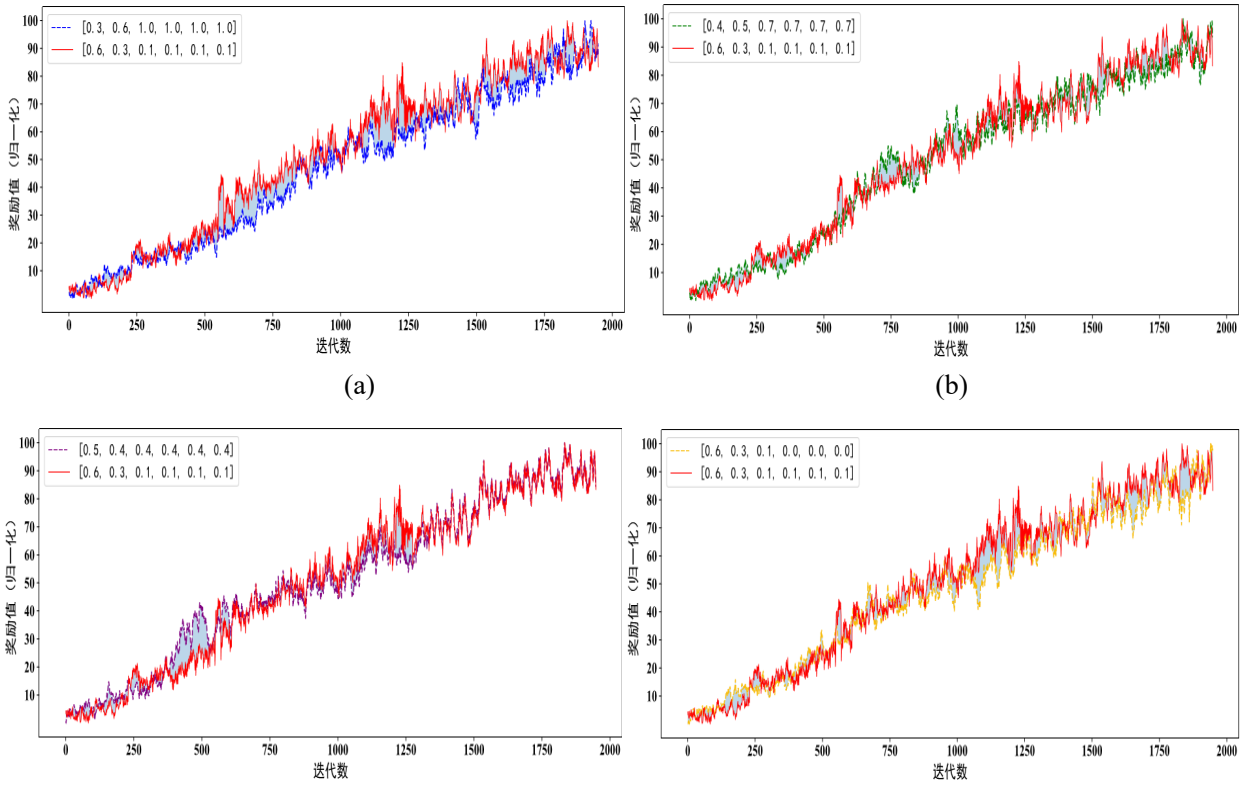


图 3-7 使用 GRU 与不使用 GRU 的对比

公式(3-1)和(3-3)都使用了权重因子用于设计流量矩阵和奖励函数的形式。这些权重因子一方面反映了每个参数指标的重要程度，另一方面决定着算法模型的收敛速度。因此，确定合适的权重因子，对于 DRL-TP 智能路由算法性能而言十分重要。首先，组成流量矩阵的权重因子对比结果如图 3-8 所示。其中，图中左上角标签中的数值分别代表L_{bw}、L_{delay}、L_{loss}、L_{used_bw}、L_{drops}和L_{errors}。从图 3-8 中四幅图可以看出，相较于其他权重因子组成的流量矩阵，权重因子分别为[0.6, 0.3, 0.1, 0.1, 0.1, 0.1]组成的流量矩阵会获得更高的奖励值，也可以发现相较于其他链路信息矩阵，L_{bw}和L_{delay}两个链路信息矩阵的权重因子值占比较大时，对 DRL-TP 智能路由算法的性能更好。因此，选择[0.6, 0.3, 0.1, 0.1, 0.1, 0.1]作为组成流量矩阵的权重因子。接下来，确定组成奖励函数的权重因子。如图 3-9 所示，图中左上角标签中的数值分别代表L_{bw}、L_{delay}、L_{loss}、L_{used_bw}、L_{drops}和L_{errors}。由于 DRL-TP 智能路由算法的目标是获取最大奖励值，因此对于奖励函数的正相关参数置为正数，负相关参数置为负数。从图 3-9 可以看出权重因子分别为[0.5, -0.4, -0.3, -0.3, -0.3, -0.3]时组成的奖励函数，相较于其他权重因子组成的奖励函数，要获得更高的奖励值。因此，选择[0.5, -0.4, -0.3, -0.3, -0.3, -0.3]作为组成奖励函数的权重因子。经过以上权重因子对比结果分析，本章实验将使用[0.6, 0.3, 0.1, 0.1, 0.1, 0.1]作为组成流量矩阵的权重因子，使用[0.5, -0.4, -0.3, -0.3, -0.3, -0.3]作为组成奖励函数的权重因子。

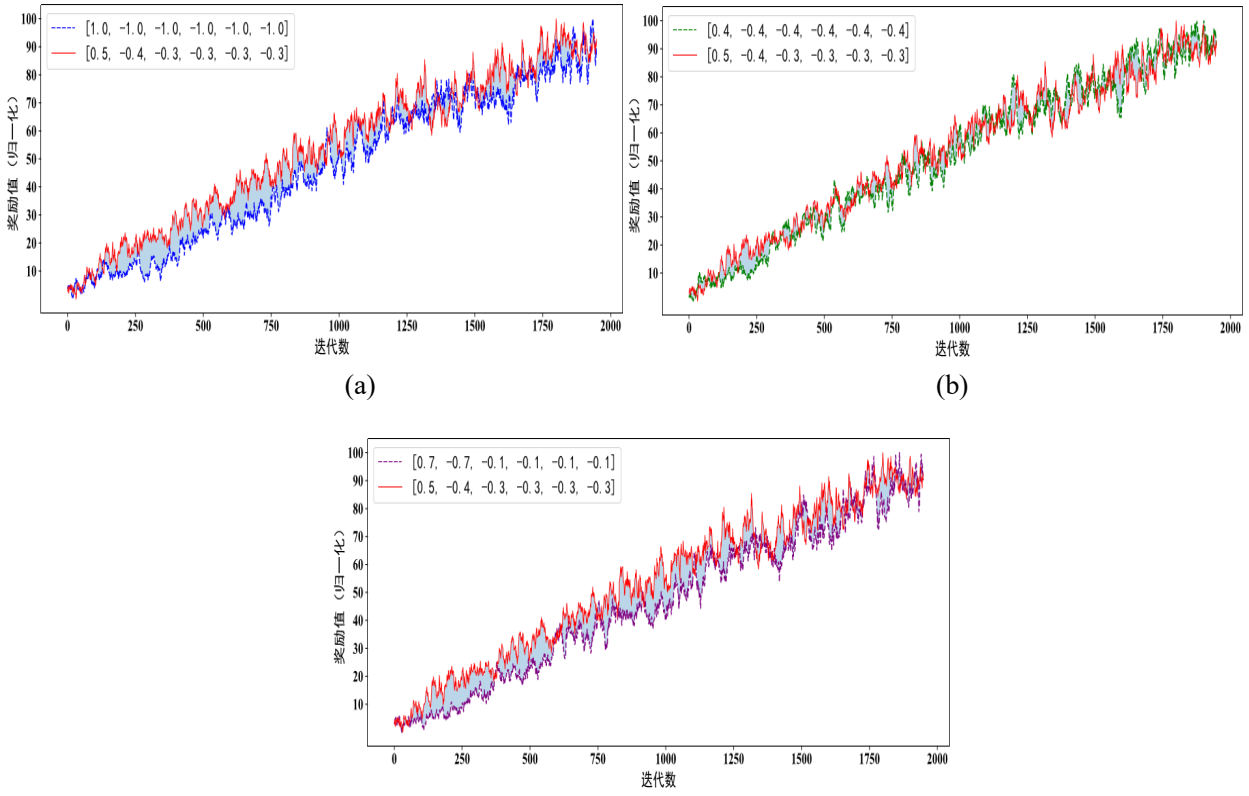
在 SDN 智能路由优化问题中，大部分研究工作都是用带宽单个网络链路指标组成的流量矩阵来构建状态空间，本章设计的 DRL-TP 智能路由算法使用了带宽、时延、



(a) (b)

(c) (d)

图 3-8 组成流量矩阵权重因子对比



(a) (b)

(c)

图 3-9 组成奖励函数权重因子对比

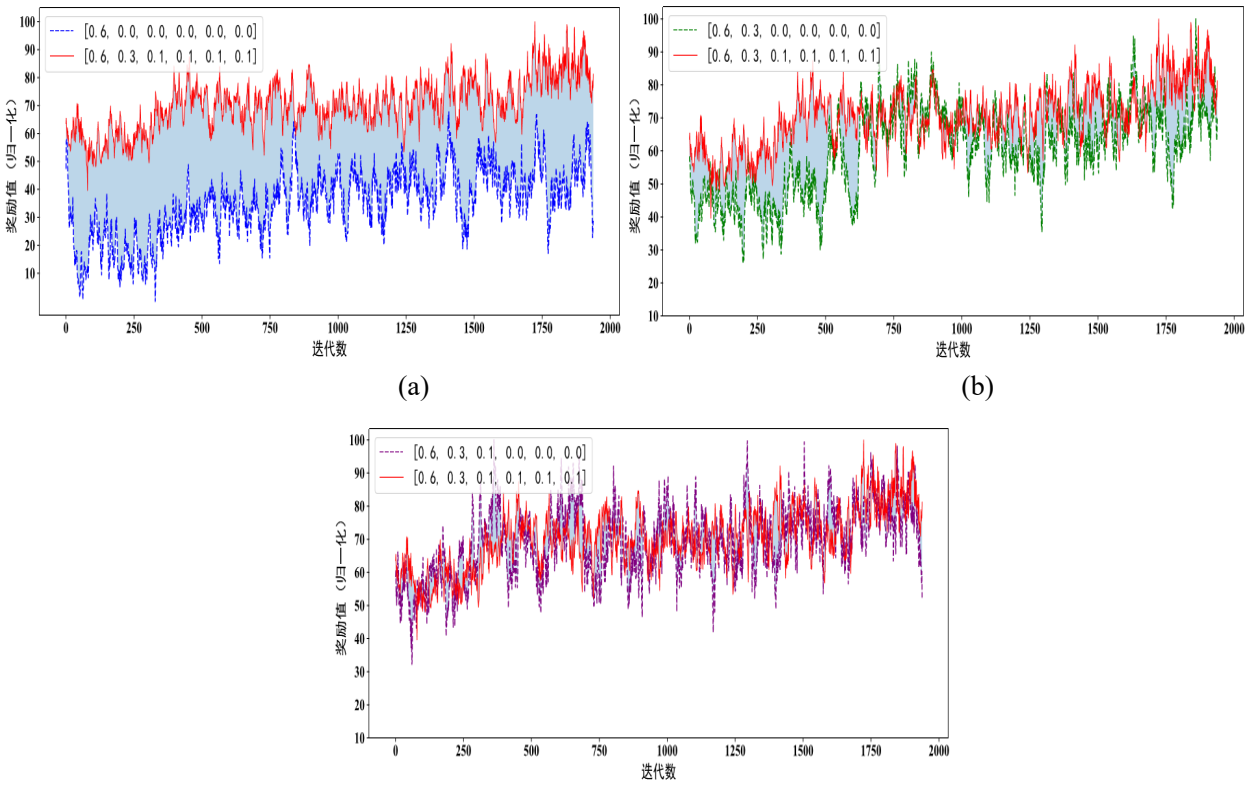


图 3-10 组成流量矩阵的网络链路指标对比

丢包率等六个网络链路指标组成的流量矩阵来构建状态空间。为了验证多个网络链路指标构建的流量矩阵，能够提升 SDN 智能路由方法的性能，本章也进行了相关实验对比如图 3-10 所示。从图 3-10(a)可以看出红色实线代表流量矩阵由L_{bw}、L_{delay}、L_{loss}、L_{used_bw}、L_{drops}和L_{errors}六个链路信息矩阵组成，明显比蓝色实线代表流量矩阵由L_{bw}单个链路信息矩阵组成，要获得更高的奖励值。图 3-10(b)可以看出绿色实线代表流量矩阵由L_{bw}和L_{delay}两个链路信息矩阵组成，性能已经比图 3-10(a)中蓝色实线代表的流量矩阵提升了很多，但总体性能还是差于红色实线代表的流量矩阵。图 3-10(c)紫色实线代表流量矩阵由L_{bw}、L_{delay}、L_{loss}三个链路信息矩阵组成，可以看出获取的奖励值已经与红色实线代表的流量矩阵相近。从三幅图对比的结果可以得出一些结论：（1）相较于使用带宽单个网络链路指标组成的流量矩阵，多个网络链路指标组成的流量矩阵，能够获得更高的奖励值，使 DRL-TP 智能路由算法表现出更加优异的性能；（2）从图 3-10(c)可以看出，提升 DRL-TP 智能路由算法的性能主要与组成流量矩阵中L_{bw}、L_{delay}和L_{loss}这三个网络链路信息矩阵有关，其他三个链路信息矩阵对 DRL-TP 智能算法的性能影响较小，但是不难发现，图 3-10(c)红色实线代表的流量矩阵的收敛速度要优于紫色实线代表的流量矩阵，原因在于多指标的权重因子能够避免某个指标过多的影响算法模型而带来的波动，从而加快算法模型的收敛速度。因此，本章使用六个链路信息矩阵用于构建流量矩阵。

§3.4.3 对比实验及结果

在 DRL 无模型算法中，可以分为基于价值和基于策略两大类。基于策略的 DRL 算法，通过概率的方式选取动作策略，对高维、连续的动作空间表现较为优异，但存在局部收敛、策略评估过程不够高效等缺点。而基于价值的 DRL 算法按最高价值选取动作策略，随着状态价值的改变，动作策略也会及时调整，因此能够较快的达到全局收敛状态，对离散的动作空间表现优异。本章中动作空间是一种离散状态由候选路径矩阵构成，加上 SDN 智能路由方法需要能够实时做出最佳路由决策。因此，在 DRL-TP 智能路由算法中，DRL 模块使用基于价值的 Dueling DQN 算法。为了验证该算法在本章实验中的性能，将其与基于策略的 DDPG 和 PPO 两种深度强化算法进行对比。如图 3-11 和图 3-12 可以看出，在训练前期，相较于 DDPG 和 PPO 两种深度强化算法，基于 Dueling DQN 的深度强化算法获取的奖励值比较低，性能表现也比较差，但随着训练次数的增加，Dueling DQN 算法性能逐渐提升，能够在较短时间内使 SDN 智能路由算法到达收敛状态。

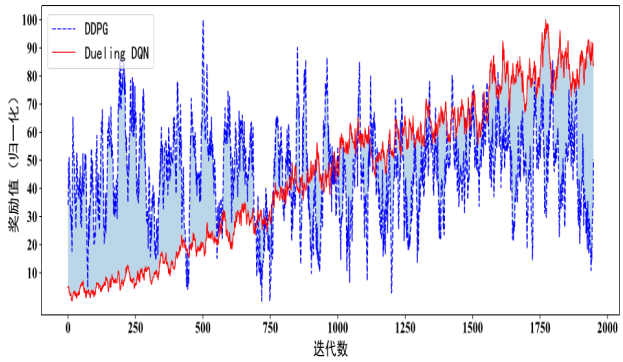


图 3-11 Dueling DQN 与 DDPG 对比

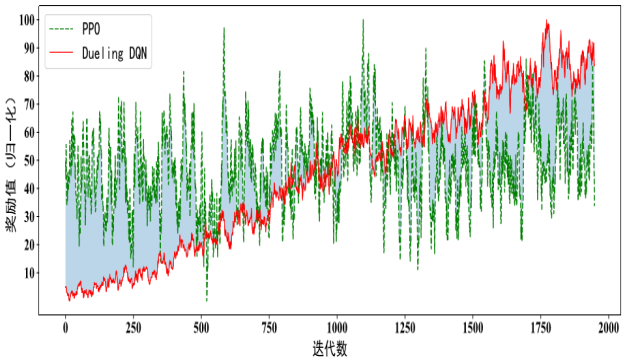


图 3-12 Dueling DQN 与 PPO 对比

本章使用迪杰斯特拉（Dijkstra）和开放最短路径优先（OSPF）两种路由算法与 DRL-TP 智能路由算法进行对比，下面分别介绍 Dijkstra 和 OSPF 两种路由算法的设计思路。

Dijkstra 路由算法：首先，在构建 SDN 网络拓扑时，对所有交换机节点之间的链

路都设置了一个链路权重值W为 1。然后，以跳数最短作为网络中每个源交换机节点到目的交换机节点的最优路由转发路径。

OSPF 路由算法：首先，使用 SDN 多线程网络测量机制获取全局网络中每条链路的时延。然后，通过链路时延得到所有源-目的交换机节点之间的路径。最后，从中选择跳数最短的路径为所有源-目的交换机节点之间的最优路由转发路径。

本章使用§2.4 小节中基于 SDN 单控制器下所设计的网络吞吐量、网络时延和网络丢包率三个网络性能评价指标，用于评价三种路由算法对网络性能产生的影响。图 3-13 为三种路由算法下的网络吞吐量对比结果。从图中可以看出，随着发送流大小的增大，三种路由算法的网络吞吐量都呈现出增长趋势，但 DRL-TP 智能路由算法的增长趋势明显大于 Dijkstra 和 OSPF 两种路由算法。

图 3-14 为三种路由算法下的网络时延对比结果。从图中可以看出 Dijkstra 路由算法随着发流大小的增大，网络时延总体呈现出指数级增长趋势。原因在于 Dijkstra 路由算法只考虑交换机之间的转发路径跳数最短，当持续增大发送流大小时，网络中相应的转发路径就会出现拥塞，而 Dijkstra 路由算法无法基于网络状态自适应的调整转发路径，导致网络中拥塞现象越来越严重，使得网络时延呈现指数级增长。当发送流大小在 10Mbit/s-40Mbit/s 时，OSPF 路由算法与 DRL-TP 智能路由算法中的网络时延相差很小，原因在于 OSPF 路由算法选择的路由转发路径是考虑链路时延指标下的跳数最短路径，它能根据网络链路时延状态的变化，动态调整相应的转发路径。因此，OSPF 路由算法下的网络时延比 Dijkstra 路由算法下的网络时延小，而且在发送流大小不是很大时，与 DRL-TP 智能路由算法下的网络时延相近。然而，随着发送流大小的不断增大，OSPF 路由算法下的网络时延逐渐高于 DRL-TP 智能路由算法中的网络时延，原因在于 OSPF 路由算法只考虑时延单个网络链路指标进行路由转发路径的选取，当网络中的流持续增大时，不能综合网络的整体指标调整路由转发策略，因此还是会出现网络拥塞的现象。而 DRL-TP 智能路由算法考虑了剩余带宽、时延、丢包率等六个网络链路指标，当网络中发送流持续增大时，可以通过相应的权重因子结合网络的整体指标，自适应智能化地调整路由转发策略，将出现拥塞的路由转发路径转移到等价的非拥塞路由转发路径，因此有效缓解了网络拥塞现象。

图 3-15 展示了三种路由算法下网络丢包率的对比结果。本章构建的 SDN 网络拓扑设置的链路带宽大小在 15Mbit-100Mbit 之间，当发送流大小在 10Mbit/s-20Mbit/s 时，SDN 网络中大多数链路都能正常进行数据包的转发，不会出现网络拥塞现象，因此，发送流大小在 10Mbit/s-20Mbit/s 时，三种路由算法的网络丢包率相差不大。但随着发送流大小不断增大，基于 Dijkstra 与 OSPF 两种路由算法选择的转发路径都出现了不同程度的网络拥塞现象，导致网络丢包率迅速增大。而 DRL-TP 智能路由算法能根据当前网络状态实时动态调整路由转发策略，选择当前网络状态下最优的路由转发

路径。因此，网络丢包率增长趋势比较缓慢，与 Dijkstra 和 OSPF 两种路由算法相比，有效的降低了网络丢包率。

从图 3-13、图 3-14 和图 3-15 可以发现，发送流大小增大到 40Mbit/s 后，网络吞吐量、网络时延和网络丢包率这三个指标都呈现出较大的增大趋势，因此可以得出以下结论：（1）基于本章构建的 SDN 网络拓扑，发送流大小增大到 40Mbit/s 后，网络中会出现较为明显的拥塞现象；（2）当网络出现较为明显的拥塞现象时，Dijkstra 与 OSPF 两种路由算法不能有效的调整路由转发策略，从而降低了网络的性能，而本章提出的 DRL-TP 智能路由算法能够实时监测网络状态，并结合多个网络链路指标动态调整最优路由转发策略，即使在网络出现较为严重的拥塞现象时，仍能保证网络的性能，验证了 DRL-TP 智能路由算法的有效性。

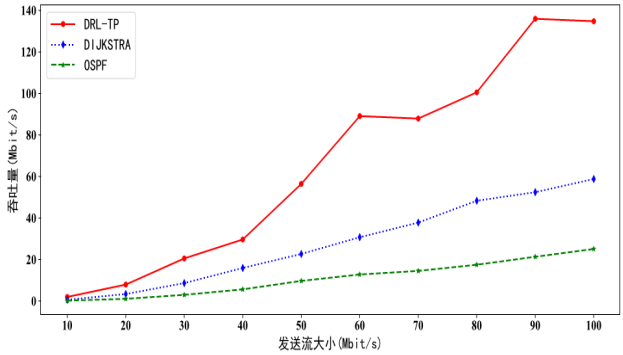


图 3-13 网络吞吐量对比

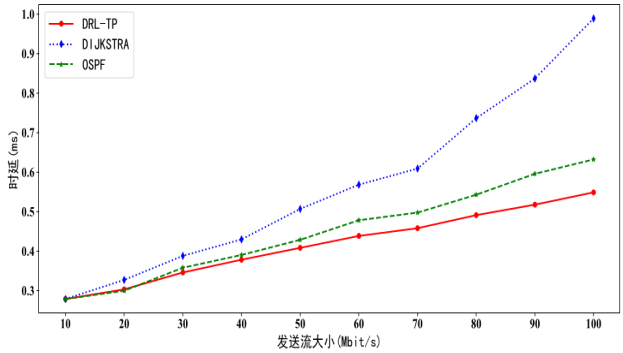


图 3-14 网络时延对比

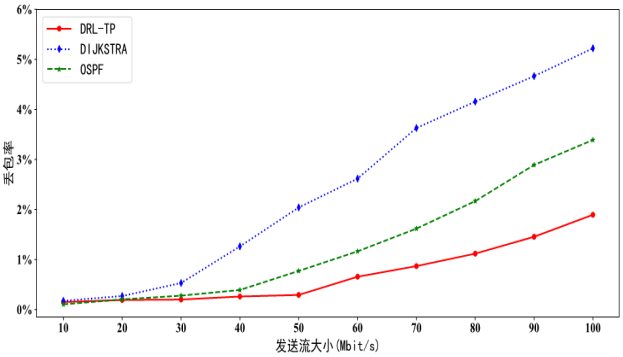


图 3-15 网络丢包率对比

§3.5 本章小结

随着 SDN 网络规模不断增大，各种新型网络设备不断出现，网络流量呈现出多样化、指数级增长等特点，如何根据 SDN 网络状态和需求找到一种实时自适应的智能路由转发策略是提升网络性能和服务质量的关键性技术。基于此本章提出了一种基于 Dueling DQN 深度强化学习和网络流量状态预测的 SDN 智能路由方法，通过设计的 SDN 多线程网络测量机制实时获取当前网络状态，并使用 DRL-TP 智能路由算法实时生成最优路由转发路径。与 Dijkstra 和 OSPF 两种路由算法相比，DRL-TP 智能路由算法在网络吞吐量、网络时延和网络丢包率都有显著提升，对解决 SDN 网络路由优化问题具有一定的实用价值。

第四章 基于多智能体的 SDN 跨域智能路由方法

本章针对大规模网络中 SDN 单控制器存在负载过大，导致网络产生数据包堆积现象而无法进行实时路由转发操作，加上传统域间路由方法存在配置繁琐和获取网络状态信息不够灵活等缺陷，导致难以获取全局网络状态信息，从而不能自适应生成最佳路由决策，提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法（MDRL-TP，**M**ulti-agent **D**eep **R**einforcement **L**earning-network **T**raffic state **P**rediction），用于解决 SDN 单控制器管理下的大规模网络中出现负载过大和流量数据包堆积等问题，实现了大规模网络中实时智能化最优路由决策的能力。本章内容分为五小节：4.1 小节给出本章的问题描述；4.2 小节对本章 SDN 多智能体跨域路由优化架构与建模进行了介绍；4.3 小节介绍了本章多智能体跨域路由算法的详解设计过程；4.4 小节给出本章的实验结果，并对结果进行了详细分析；4.5 小节对本章的研究工作进行了总结。

§4.1 问题描述

§4.1.1 SDN 多控制器架构

基于 SDN 单控制器管理下的智能路由优化算法通过灵活获取全局网络状态信息，并能根据当前网络状态信息进行实时最优路由决策，解决了传统路由方法仅使用有限的网络状态信息进行路由决策的缺陷，显著提升了网络性能。然而，随着网络规模的增大，网络中将存在大量的数据流，基于 SDN 单控制器管理下的大规模网络面临着控制器负载过大等问题，难以保证数据包得到及时有效的处理，导致无法实时获取全局网络状态的信息，从而影响网络的性能。因此，为了解决 SDN 单控制器管理下的大规模网络中出现的性能瓶颈问题，基于 SDN 多控制器架构下的分域管理已经成为一种关键性技术。

目前，大部分研究工作[73][74][75][76]中主要将 SDN 多控制器架构分为两种形式：扁平式架构和分层式架构。在扁平式架构中，每个控制器都具有相同等级，相邻控制器之间通过彼此通信来交换信息，进而获取全局网络状态信息。分层式架构一般包含两层，第一层由根控制器组成，第二层由本地控制器组成。本地控制器负责本域的信息交互和路由决策，根控制器协同控制器之间的通信，负责域间的信息交互和路由决策。扁平式架构是对 SDN 控制平面功能的扩展，需要复杂的控制管理和开销。例如，控制器之间需要频繁通信以保证网络全局视图的一致性；某个控制器下的网络拓扑发生

变化时，控制器之间需要重新交互以获取全局网络拓扑信息。而分层式架构能够并发获取多个控制器管理下的网络链路信息，因此获取全局网络链路信息的速度比较快。考虑到本章主要是解决大规模网络中 SDN 多控制器智能路由优化的问题，需要实时获取全局网络链路信息，基于两种架构的特性，决定采用分层式架构。

基于 SDN 多控制器分层式架构管理下的大规模网络中必须要解决控制器之间的消息转递和消息同步等关键性问题。传统边界网关协议（BGP，border gateway protocol）[77]是利用边界路由器进行相邻自治系统（AS，autonomous system）之间的消息转递，以实现域间消息同步。但基于 BGP 方式实现消息传递与消息同步在 SDN 环境中配置较为繁琐且存在路由震荡等问题。OpenFlow1.3 协议[78]提供了一种东西向接口以实现 SDN 跨域路由消息的传递。然而，与南北向接口标准化的广泛使用相反，东西向接口还没有形成业内一致认可的标准。对此有部分学者已经在 SDN 东西向接口的工作中[79][80][81]取得了一些成果，通过自适应方式更新控制器之间的状态，实现全局网络状态的一致性，有效解决了多控制器间消息传递和消息同步等问题。然而，基于自适应方式实现多控制器间的消息传递和消息同步，存在适应时间较长、收敛速度慢等问题，难以达到实时获取全局网络状态信息的目的。此外，如何保证多控制器间消息的可靠性和稳定性传输，也是确保 SDN 多智能体路由优化方法性能的关键。

§4.1.2 多智能体深度强化学习机制

凭借强化学习在解决序贯决策问题和深度学习在解决高维、复杂问题上的优势，基于单智能体的 DRL 算法已经在流量工程、负载均衡、路由优化和自动驾驶等领域取得了许多成果。然而，随着网络规模不断增大，基于单智能体的深度强化学习面临着负载大、收敛难等问题，不利于解决大规模网络路由优化问题。近年来，DRL 在多智能体系统中取得了重大进展，加上分布式技术的不断成熟，许多学者都已经开始对多智能体深度强化学习机制（MDRL，Multi-Agent Deep Reinforcement Learning）展开研究并在博弈对抗、无人机集群以及网络数据中心等领域取得了一定的成果[82][83][84]。目前，多智能体的深度强化学习机制主要分为：集中学习、独立学习和 CTDE（Centralized Training Decentralized Execution）三种方式。

集中学习是将所有的智能体当作一个整体进行学习，即将每个智能体的状态和动作空间结合生成联合状态和动作空间，它整合了所有智能体的状态信息，因此较为容易的获取全局最优解。但是，随着智能体数量的增加，状态和动作空间的维度将会急剧增大，而集中学习方式无法利用分布式训练的优势，导致智能体产生巨大探索和试错的时间成本开销，使得模型训练时间过长。此外，基于 MDRL 机制下的集中式学习方式，每次训练都要获取网络全局状态信息，不太符合实际应用场景。

独立学习简单来讲就是将多个智能体进行独立的堆叠部署，每个智能体都拥有自己的环境，与其他智能体互不干扰，在训练的过程中独自更新各自的网络模型，根据各自最大化奖励值的目标进行最优策略的选取。它的优点是不需要考虑各个智能体之间的协作与交互，实现较为简单，能够使用分布式技术加速多智能体的训练时长，适用于离散型状态和动作空间较小的问题。但是，独立学习方式需要智能体都处于相对平衡的环境中，一旦环境处于动态变化中，智能体的稳定性和收敛性都难以保证。

CTDE 是集中学习和独立学习两种方式的结合，它融合了集中学习和独立学习方式的优点，使用分布式技术加快多智能体的模型训练时长，允许智能体根据相应需求利用全局状态信息进行集中学习训练，在训练结束后，进行多智能体的分布式最优决策，它是一种比较常见的多智能体强化学习方式。

针对以上问题，本章利用所设计的 SDN 多线程网络子域测量机制实时获取每个网络子域中的流量矩阵，并通过协同通信模块中的 socket[85]技术用于多控制器之间点对点的通信，实现域间消息传递与消息同步，提升了获取全局网络状态信息的收敛速度并保证了多控制器之间消息的可靠性和稳定性传输，进而实时获取全局网络中的流量矩阵，并通过本地控制器和根控制器中的智能体分别实时自适应生成当前网络状态下的域内和域间最优路由转发路径，以解决 SDN 单控制器管理下大规模网络中出现负载过大和流量数据包堆积等问题。最后，由 MDRL-TP 多智能体跨域路由算法实时生成全局网络的最优路由转发路径。

§ 4.2 SDN 多智能体跨域路由优化架构与建模

本章设计的 SDN 多智能体跨域路由优化架构如图 4-1 所示，主要包括根控制器、本地控制器以及协同通信模块。本章中的 SDN 多智能体跨域路由优化架构是对第三章中的 SDN 智能路由优化框架的扩展。因此，在 SDN 多控制器上也部署了数据平面、控制平面、管理平面和知识平面，本章仅介绍在这四个平面上新增和不同的功能，其他相同功能已在第三章中介绍。

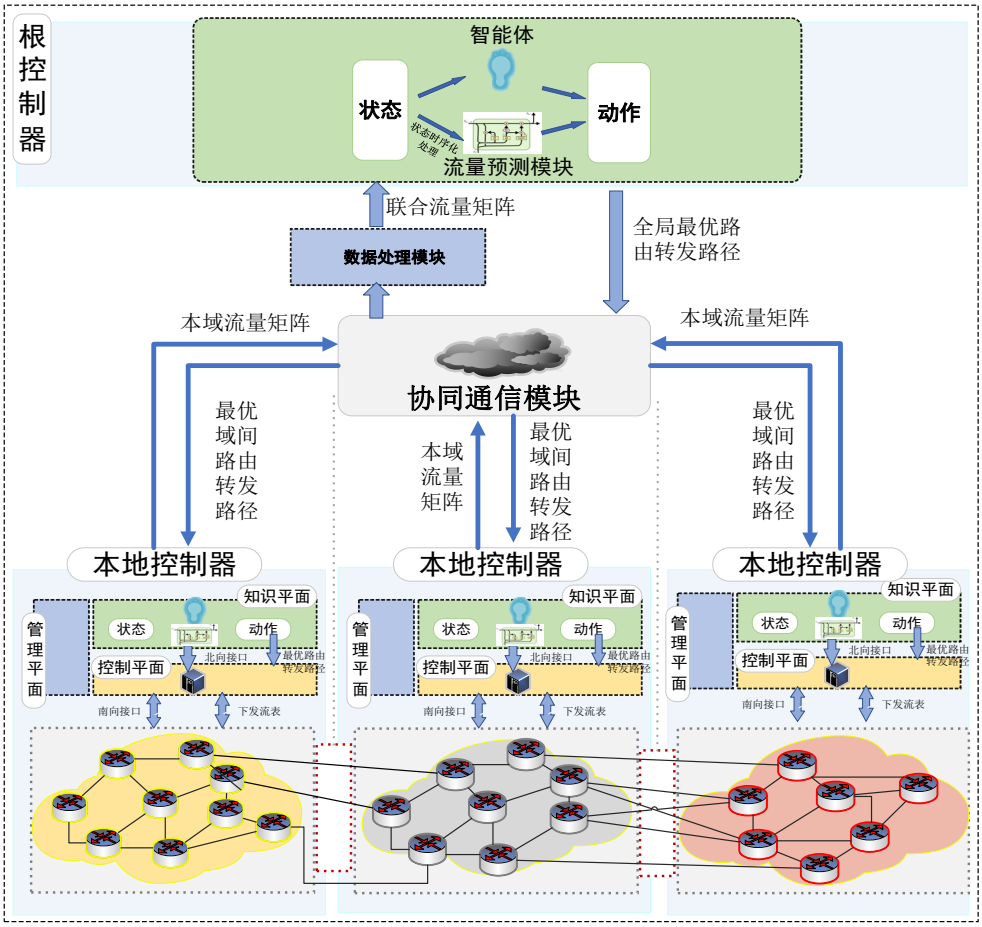


图 4-1 SDN 多智能体跨域路由优化架构

§4.2.1 根控制器

根控制器作为 SDN 多智能体跨域路由优化架构的中枢，它主要负责获取全局网络的链路状态信息和生成最优域间路由转发路径。因此，在根控制器中部署了管理平面和知识平面。

管理平面实现的功能主要包括（1）将获取的多个网络子域中的链路状态信息存储到信息池中；（2）将信息池中的网络链路信息转换成联合流量矩阵，提供给根控制器中的知识平面进行模型线下训练。

知识平面主要包括 DRL 模块和网络流量状态预测模块。DRL 模块，即智能体，主要使用管理平面提供的联合流量矩阵来构建网络训练环境，并通过具有高维决策的 DRL 模型与构建环境进行持续交互以获得更高的奖励。当模型迭代趋于收敛后，给出当前时刻网络状态的动作，即为最优路由转发路径。知识平面要实时获取全局网络状态信息以做出路由决策，这就需要每个网络子域中的 SDN 控制器频繁、不间断响应处理本域中的数据流。然而，这种响应处理机制会使每个网络子域中的 SDN 控制

器负载过高、消耗过大，存在一定的宕机风险，致使网络中一些数据流无法及时处理，导致网络出现波动，影响SDN网络的正确运行。因此，本章对第三章中的SDN多线程网络测量机制进行了改进，设计了一种间隔式的SDN多线程网络子域测量机制用于获取每个网络域中的链路状态信息，不仅有效解决了每个网络子域中的SDN控制器在连续处理大量、不同类型数据流时带来的高负载和消耗大的问题，而且还能较好的满足根控制器中的知识平面实时获取联合流量矩阵的需求，但还是会存在对每个网络子域中一些域内流量矩阵监测的遗漏，从而影响多智能体跨域路由方法的性能。为了解决间隔式SDN多线程网络子域测量机制对网络域中的一些域内流量矩阵存在监测遗漏的问题，在每个知识平面中都添加了一个网络流量状态预测模块，用于预测出监测遗漏的流量矩阵和发现网络中隐藏和未知的网络流量状态。

本章使用分层式架构用于搭建SDN多控制器网络，能够并发的获取每个控制器管理下的网络链路信息，并以较快的速度获取全局网络链路信息。但是，存在大规模网络中高量数据流量请求导致根控制器负载过大等问题，解决思路是可以使用根控制器扩容或分流等机制，但需要花费较大的硬件资源。考虑到实验条件和实际的应用场景，本章采用的是请求间隔式响应机制来解决根控制器负载过大的问题。具体而言，每个网络子域中的最优域内路由转发路径由本域中的知识平面进行路由决策生成，而最优域间路由转发路径由本地控制器请求根控制器中的知识平面进行路由决策生成，这样就有效解决了大规模网络中大量网络流量请求根控制器进行路由决策而带来负载过大的问题。同时，考虑到DRL算法需要不断的进行探索和试错才能保证智能体采取最优决策动作。因此，根控制器除了被动响应域间请求进行路由决策外，还会以主动间隔式的方式获取全局网络的链路状态信息进行智能体的学习，以保证根控制器中的智能体能够生成最优路由决策。

§4.2.2 本地控制器

本章将整个网络划分为三个子域，每个本地控制器管理一个子域。本地控制器实现的主要功能有:（1）处理本域内的路由转发请求并生成最优域内路由转发路径;（2）发送域间路由请求给根控制器，收到根控制器响应的最优域间路由转发路径后，生成最优域间路由转发策略。在每个网络子域中都部署了数据平面、控制平面、管理平面和知识平面。

数据平面实现的主要功能有:（1）通过周期性响应控制平面发送的相应请求提供本域网络中全局的感知信息，例如：例如交换机端口速率、发送数据包数量和接收字节数等等;（2）智能体生成最优域内路由转发路径，经控制平面生成最优路由转发策略后，由数据平面根据最优路由转发策略对数据包进行转发操作。

控制平面主要部署了两个模块：网络信息感知模块和路由转发模块。网络信息感

知模块的主要功能为：（1）使用 LLDP 协议获取当前网络子域中的拓扑信息；（2）通过使用相应的 Request 指令，周期性查询每个网络子域中所有交换机的详细状态信息，由此生成本域中的感知信息。路由转发模块主要分为域内和域间转发，域内转发的主要功能包括：（1）由本地控制器中的知识平面生成最优域内路由转发路径，根据本域中的网络拓扑找到对应的主机节点；（2）根据最优域内路由转发路径和对应的主机节点，生成最优域内路由转发策略并下发到数据平面。域间转发的主要功能包括：（1）将域间路由转发请求发送到根控制器中；（2）根控制器通过知识平面生成最优域间路由转发路径并响应给对应的本地控制器；（3）本地控制器根据最优域间路由转发路径和对应的主机节点，生成最优域间路由转发策略并下发到数据平面。

§4.2.3 协同通信模块

目前，SDN 多域之间的通信和协同技术比较常用的是 BGP 协议和 SDN 东西向接口。BGP 是一种用于大规模数据中心下维护不同 AS 之间路由信息的、无中心的路由协议，主要用于交换 AS 之间的可达路由信息，从而构建 AS 域间传播路径，实现多个网络域之间的通信和路由决策。它被广泛应用于传统网络中不同网络域之间的路由决策，但存在改变路由策略时引发路由振荡的问题，而且随着网络的扩大，路由条目越多，配置和管理工作也就越复杂。SDN 东西向接口是通过 SDN 东西向协议，实现多个网络子域之间通信与协作。然而，东西向接口目前还处于研究初级阶段，而根控制器需要一种可靠机制用于传输每个本地控制器中的网络状态信息，进而实时获取全局网络状态信息。因此，基于 SDN 东西向接口很难满足这样的需求。

本章使用 socket 技术设计了一个协同通信模块，用于实现 SDN 多域之间的消息传递与同步。socket 是对网络中不同主机上的应用进程之间进行双向通信的端点抽象，是 TCP/IP 通信协议的基本操作单元。因此，socket 是一种可靠的传输机制。socket 通信过程十分简便，客户端通过服务器绑定的 IP 地址和端口进行连接，实现客户端和服务端双方的通信与网络信息传输。本章将根控制器作为服务端，本地控制器作为客户端，它们之间的消息体被封装在 Response 和 Request 数据结构中，并通过 JSON 格式进行传递。协同通信模块的主要功能包括：（1）本地控制器通过 socket 向根控制器发送最优域间路由转发路径的请求；（2）根控制器通过 socket 向本地控制器响应最优域间路由转发路径；（3）根控制器主动间隔式的通过 socket 获取每个网络子域的流量矩阵用于组成联合流量矩阵，从而实现获取全局网络链路状态信息。

协同通信模块能否快速、高效的将本地控制器的请求传给根控制器、根控制器的响应传给本地控制器以及将每个网络子域的流量矩阵传给管理平面，决定了 SDN 多智能体路由算法的有效性。因此，本文使用了多线程和管道技术，用于保证协同通信

模块中的 socket 能够高效处理请求和响应以及获取每个网络子域中的流量矩阵。具体而言，分别创建了三个线程用于处理本地控制器的请求、根控制器的响应和每个网络子域流量矩阵的获取。其中，在每个本地控制器管理的网络子域中，获取本域的流量矩阵采用的是 SDN 多线程网络子域测量机制，如图 4-2 所示。主要流程为：创建信息探测线程，用于每隔t秒向本域控制平面请求监测当前时刻下的本域网络感知信息，本域控制平面收到监测请求后，通过发送相关的 Request 请求指令将获取到的本域网络感知信息返回。然后，将本域网络感知信息传入数据处理线程，通过相关公式转换成带宽、时延、丢包率等网络链路信息并构建本域中的域内流量矩阵。数据处理线程是 SDN 多线程网络子域测量机制的核心，它基于跨域智能路由方法中训练测试和应用部署两种模式分别执行两种不同的职能。如图 2 蓝色箭头序号是训练测试模式下，主要通过信息操作线程将每个时刻构建的域内流量矩阵保存到信息池中用于模型离线训练；图 4-2 中的红色箭头是应用程序部署模式下，主要将域内流量矩阵输入到训练模型中，由本网络域中的知识平面生成最优域内路由转发路径，然后通过控制平面生成最优域内路由转发策略。最后，根据相关等式计算得到本域内的网络吞吐量、时延、丢包率三个评价指标。

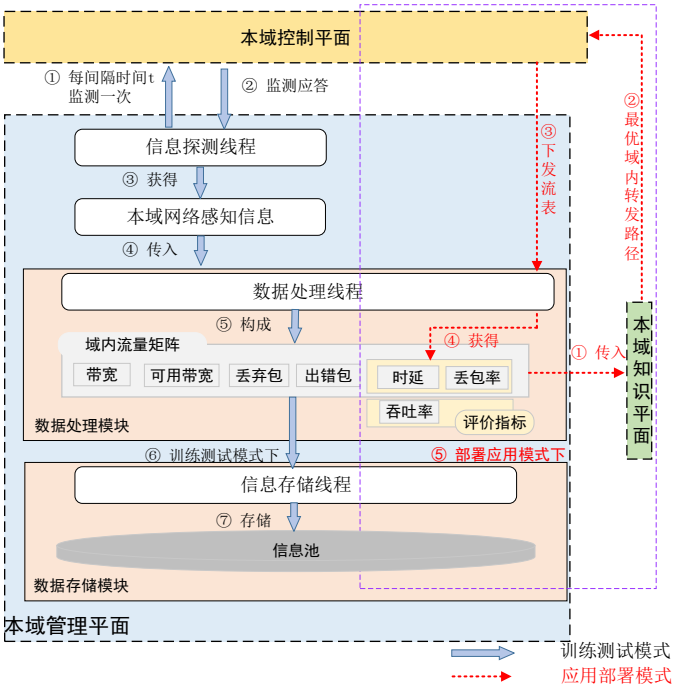


图 4-2 SDN 多线程网络子域测量机制

由于 socket 是一种受限的资源，频繁进行根控制器与本地控制器之间的通信，特别是对于根控制器主动间隔式获取每个网络子域中的流量矩阵，难免会带来一定的资源和时间开销。因此，为了减少在获取每个网络子域中的流量矩阵时与 socket 交互的次数，同时增大每次 socket 通信时传输的数据大小，采取了管道（Pipe）技术，它本质上就是一个缓冲区，当缓冲区中数据大小满足一定条件时（本章设置为 1024 字节），

将缓冲区的数据进行批次传输。

在本章根控制器与本地控制器之间进行消息传递和同步过程中，涉及到一些设置、标识和确认的操作，例如：标识每个本地控制器的名称、标识哪些是处理请求的线程、哪些是处理响应的线程等等。为了能够更加方便的进行上述操作，本章自定义了一些状态码，它的符号和代表的含义如表 4-1 所示。

表 4-1 状态码信息

状态码	状态符号	状态含义
0001	SET_CONTROLLER_NAME	标识本地控制器的名称
0010	ADD_DOMAIN_TOPO	标识网络拓扑加入到相应本地控制器中
0011	SYN_GLOBAL_VIEW	标识获取全局网络的链路状态信息
0100	ADD_INTER_DPID	确认当前网络中的域间交换机，用于寻找域间主机
0101	REQ_INTER_ PROPERTY	标识本地控制器请求最优域间路由转发路径
0110	RES_INTER_ PROPERTY	标识根控制器响应最优域间路由转发路径

§ 4.3 MDRL-TP 多智能体跨域路由算法设计

本章中的网络拓扑是一个无向图，可以用$G = \{D，V，E，W\}$表示。其中，D代表当前网络子域，V代表数据平面中 SDN 交换机节点，E代表交换机节点之间的链路，W代表链路的权重大小，一般设置为常量。MDRL-TP 多智能体跨域路由算法总体流程如图 4-3 所示。

首先，三个网络子域中最优域内转发路径分别由三个本地控制器中的智能体 1、2 和 3 生成，具体流程为：（1）通过 SDN 多线程网络子域测量机制获取本域中的流量矩阵，由本域中的网络流量状态预测算法获取预测流量矩阵，共同组成 DRL 算法的状态空间；（2）通过线下的方式训练本地控制器中的 DRL 和预测模型，并组成当前网络子域中的 DRL-TP 算法模型；（3）根据当前发送流大小获取网络子域中的流量矩阵，通过 DRL-TP 算法实时生成最优域内路由转发路径。

然后，根控制器中的智能体通过主动间隔式获取全局网络链路状态信息进行模型

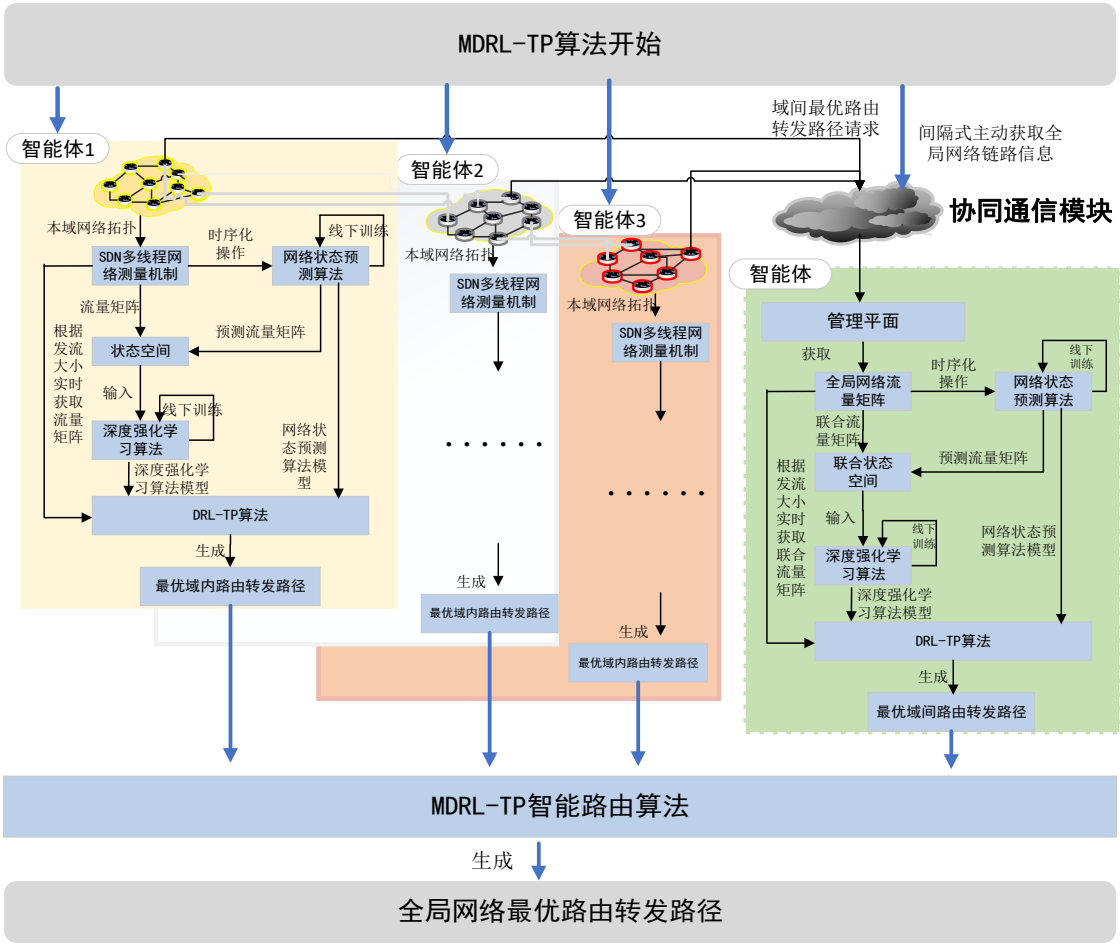


图 4-3 MDRL-TP 多智能体跨域路由算法总体流程

线下训练和被动响应每个网络子域中最优域间路由转发请求生成最优域间路由转发路径，具体流程为：（1）通过协同通信模块获取每个网络子域中的流量矩阵，由管理平面进行数据操作后生成联合流量矩阵，由网络流量状态预测算法获取预测联合流量矩阵，共同组成联合状态空间；（2）通过线下的方式训练根控制器中的 DRL 和预测模型，并组成 DRL-TP 算法模型；（3）根据每个网络子域中的最优域间路由转发请求，由根控制器中的智能路由算法生成最优域间路由转发路径。

最后，根据所得的最优域内和域间路由转发路径，经 MDRL-TP 多智能体跨域路由算法生成全局网络中最优路由转发路径。下面分别对设计的 Dueling DQN 深度强化学习算法、网络流量状态预测算法和 MDRL-TP 多智能体跨域路由算法进行介绍。

§4.3.1 Dueling DQN 深度强化学习算法

本章使用 CTDE 方式作为多智能体的深度强化学习机制，它融合了集中学习和独立学习两种方式的优点。如图 4-4 所示，三个本地控制器中的智能体 1、2 和 3 分别采用独立学习的方式，在训练中独自更新自己的网络模型进行最优策略选取，生成

本域中最优域内路由转发路径。根控制器中的智能体采用集中学习方法，可以表示为公式(4-1)：

$$(N, US, UA, P, \Upsilon, UR) \tag{4-1}$$

其中，N表示智能体数量；$US = S_{D_1} \cup S_{D_2} \cup S_{D_3}$表示多智能体的联合状态空间,其中$S_{D_d}$ $(d = 1,2,3)$为三个网络子域的状态空间；$UA = \{ua_1, ua_2, \ldots, ua_n\}$表示多智能体的联合动作空间 $(n = |V_{D_1} \cup V_{D_2} \cup V_{D_3}| \cdot |V_{D_1} \cup V_{D_2} \cup V_{D_3}|$，其中$|V_{D_1} \cup V_{D_2} \cup V_{D_3}|$为整个网络拓扑中交换机节点的数量)；$P$表示状态转移函数；$\Upsilon$表示折扣因子；$UR$为联合奖励函数，表示根控制器中的智能体在$t$时刻下，联合状态$us_t \in US$执行联合动作$ua_t \in UA$获取的联合奖励值$ur_t$。

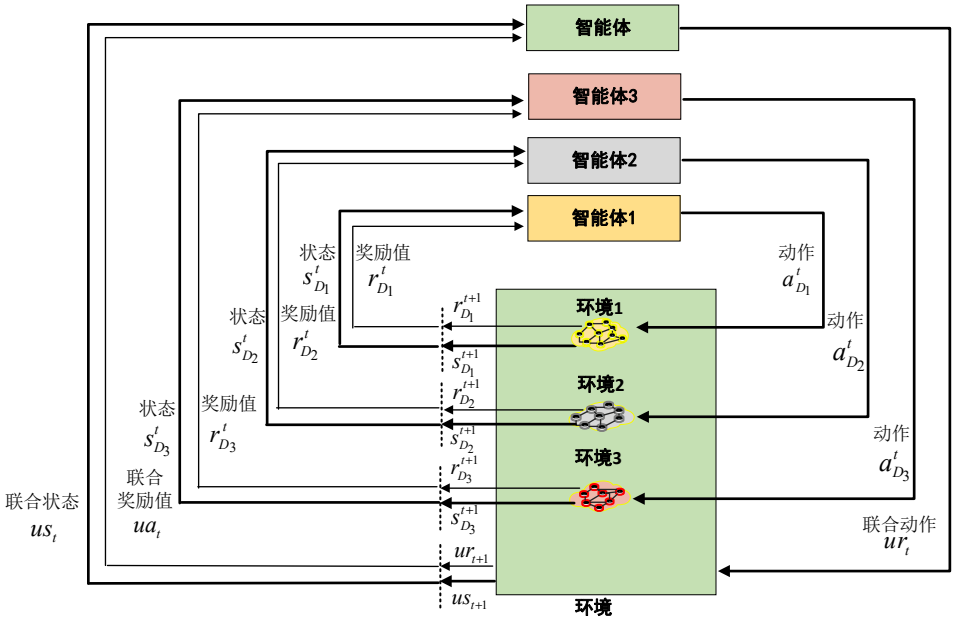


图 4-4 基于多智能体的强化学习机制

DRL 是一种框架性算法，针对不同的决策问题需要设计不同的状态、动作和奖励函数。本章中三个本地控制器中的智能体 1、2 和 3 的状态空间$(S = TM)$、动作空间(A)和奖励函数(R)的设计与 § 3.3.1 小节中深度强化学习算法的设计思路基本一致。因此，本章仅介绍根控制器中智能体的联合状态空间、联合动作空间和联合奖励函数的设计。

联合状态空间（US）。联合状态空间可表示为$US = UTM$，其中UTM指在间隔t内的联合流量矩阵由多个二维矩阵$M_{|V_{D_1} \cup V_{D_2} \cup V_{D_3}| \times |V_{D_1} \cup V_{D_2} \cup V_{D_3}|}$拼接而成，如公式(4-2)所示：

$$um_{ij} = w_1 \cdot \frac{1}{Lbw_{ij}} + w_2 \cdot L_{delay_{ij}} + w_3 \cdot L_{loss_{ij}} + w_4 \cdot L_{used_{bw_{ij}}} + w_5 \cdot L_{drops_{ij}} + w_6 \cdot L_{errors_{ij}} \quad , i, j = 1, 2, .., |V_{D_1} \cup V_{D_2} \cup V_{D_3}| \tag{4-2}$$

其中，um_{ij}表示联合矩阵中的元素，由L_{bw}、L_{delay}、L_{loss}、$L_{used_{bw}}$、L_{drops}和 L_{errors} 6 个方面的网络链路剩余带宽、时延、丢包率、已用带宽、弃包数和错包数信息矩阵元素通过相加映射组成，每个网络链路信息矩阵包含了当前时刻下全局网络中所有交换机节点之间的链路信息；D_1, D_2, D_3表示三个网络子域；$w_l \in [0,1], l = 1,2,..,6$表示可调参数，为构成联合流量矩阵元素的权重因子；$i$和$j$表示全局网络拓扑中交换机节点名称；$\left|V_{D_1} \cup V_{D_2} \cup V_{D_3}\right|$表示整个网络拓扑中交换机节点的数量。联合流量矩阵结构图如图 4-5 所示。

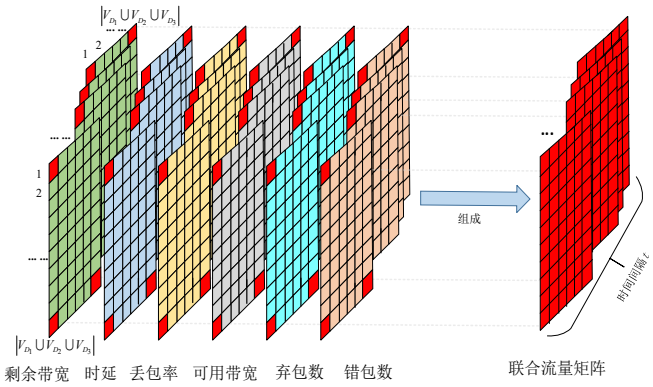


图 4-5 联合流量矩阵结构图

考虑到组成联合流量矩阵的每个信息矩阵中的元素值存在较大差异，无法客观反映出各个网络链路信息矩阵对所组成联合流量矩阵的贡献值，使得 MDRL-TP 多智能体跨域路由算法难以收敛，从而影响网络模型的性能。因此，本章同样使用 Min-Max 技术将联合流量矩阵中的元素归一化到区间$[\mu_1, \mu_2]$范围内，以提升网络模型的收敛速度和性能，如公式(4-3)所示：

$$\overline{um_{ij}} = \mu_1 + \frac{\left(um_{ij} - \min\left(UTM\right)\right) \bullet \left(\mu_2 - \mu_1\right)}{\max\left(UTM\right) - \min\left(UTM\right)} \tag{4-3}$$

其中，$\overline{um_{ij}}$表示联合流量矩阵归一化后的元素；$\min(UTM)$和$\max(UTM)$分别表示联合流量矩阵中最小和最大的元素。

联合动作空间（*UA*）。动作空间是智能体根据当前网络状态生成的行为，即智能体所采取的路由决策。本章设计的联合动作空间是一种离散型的，每个联合动作$ua_t \in [0,1,\ldots,k]$对应于当前t时刻下，联合状态$us_t \in US$下的路由转发路径选择，由$k = \left\lceil 0.1 \cdot \max\left(\left|V_{D_1}\right|, \max\left(\left|V_{D_2}\right|, \left|V_{D_3}\right|\right)\right) \cdot \max\left(\left|V_{D_1}\right|, \max\left(\left|V_{D_2}\right|, \left|V_{D_3}\right|\right)\right)\right\rceil$个候选路径矩阵$C_{\left|V_{D_1} \cup V_{D_2} \cup V_{D_3}\right| \times \left|V_{D_1} \cup V_{D_2} \cup V_{D_3}\right|}$组成，每个候选路径矩阵中包含了所有源交换机节点到目的交换机节点的路径。其中，每个候选路径矩阵中的元素为交换机节点i到交换机节点j的路径$path_{ij} = [i, \ldots, j]$。

联合奖励函数（*UR*）。奖励是智能体执行相应动作后产生的收益指标，用于引导智能体采取更优的动作策略以获得更高的收益回报。MDRL-TP 多智能体跨域路由算

法模型中，奖励函数如公式(4-4)所示，优化的目标是最大化剩余带宽，最小化时延、丢包率、已用带宽、弃包数及错包数。而深度强化学习以不断获取最大奖励值为目标，因此对于最大化指标设置正相关系数 1，对于最小化指标设置负相关系数-1。$\varphi_l \in [0,1], l = 1,2,\ldots,6$表示可调参数，为构建联合奖励函数的权重因子。

$$UR = \varphi_1 \bullet L_{bw} - \varphi_2 \bullet L_{delay} - \varphi_3 \bullet L_{loss} - \varphi_4 \bullet L_{used_{bw}} - \varphi_5 \bullet L_{drops} - \varphi_6 \bullet L_{drops} \tag{4-4}$$

本章使用 Dueling DQN 作为深度强化学习算法，其中，价值函数如公式(4-5)所示，ω表示价值函数VF和优势函数AF共有的网络参数;β, λ分别表示VF和AF专有的网络参数。同时，MDRL-TP 多智能体跨域路由算法也使用了衰减 ε-贪婪探测机制，如公式(4-6)所示。

$$Q(us, ua, \beta, \lambda) = VF(us, \omega, \beta) + AF(us, ua, \omega, \lambda) \tag{4-5}$$

$$ua_t = \begin{cases} argmax_{ua} Q_{policy}(\Phi(us_t), ua, \theta), if\ x < 1-\varepsilon \\ random.random(), \quad otherwise \end{cases} \tag{4-6}$$

下面给出本章中 Dueling DQN 深度强化学习算法具体实现的细节，如算法 4-1 所示。其中，联合流量矩阵UTM由协同通信模块收集的域间流量矩阵、每个网络子域内的流量矩阵TM和网络流量状态预测模块生成的预测流量矩阵组成。步骤 1：生成结构相同的策略网络Q_{policy}和目标网络Q_{target}以及存放样本的缓冲池M。步骤 3：智能体获取初始联合状态us_t。步骤 6-步骤 8：选择当前联合状态us_t下的联合动作ua_t，并获取相应的联合奖励回报ur_t，然后将经验样本存放到M中。步骤 9-步骤 13：进行策略网络Q_{policy}的参数更新操作。步骤 14-步骤 16：当训练步长$steps$达到更新频率$freq$时，将网络Q_{policy}的权重和偏差更新到网络Q_{policy}。步骤 17：将智能体移动到下一个联合状态。每次训练结束时，输出当前网络状态下所有源-目的交换机节点的转发路径。

算法 4-1 Dueling DQN 深度强化学习算法

输入:

学习率: lr

采样大小: $batch$

折扣因子: γ

权重因子: $\varphi_l \in [0,1], l = 1,2,\ldots,6$

衰减参数: ε

衰减率: $decay$

目标网络更新频率: $freq$

训练总数: $episodes$

联合流量矩阵: UTM

输出:

网络中所有源-目的交换机节点的转发路径

初始化Q_{policy}及Q_{target}网络权重θ、经验池M
For $episodes \leftarrow$ 1 to n do:
  智能体获取网络环境初始联合状态us_t
  While next union state us_{t+1} is not final state do:
    更新衰减参数 $\varepsilon = \varepsilon - (steps \cdot decay)$
    根据等式(4-6)智能体获取当前联合状态us_t下的联合动作ua_t
    根据等式(4-3)智能体获取当前联合奖励值$ur_t \leftarrow UR(us_t, ua_t)$
    将经验Experiences$_t = (us_t, ua_t, ur_t, us_{t+1})$存放到$M$
    If len$(M) \geq batch$ then:
      从M中随机采样大小数据
      根据价值函数等式(4-5)分别获取p_value以及t_value
      $p\_value = Q_{policy}(us_t, ua_t, \theta)$
      $t\_value = \begin{cases} ur_t, & if\ next\ union\ state\ is\ final\ state \\ ur_t + \gamma \cdot max_{ua'} Q_{target}(us_{t+1}, ua'; \theta), & otherwise \end{cases}$
      使用$loss = (t_value - p_value)^2$执行梯度下降更新$Q_{policy}$ 网络权重参数θ
    End if
    If $steps \,\%\, freq == 0$ then:
      更新Q_{target}网络模型参数，$\theta^{Q_{target}} \leftarrow \tau \cdot \theta^{Q_{policy}} + (1-\tau) \cdot \theta^{Q_{target}}$
    End if
    $us_t \leftarrow us_{t+1}$
  End while
End for

§4.3.2 网络流量状态预测算法

本章使用 GRU 作为网络流量状态预测算法，如算法 4-2 所示。步骤 1：初始化 GRU 网络权重参数。步骤 2：将时序化操作后生成UTM_{input}和UTM_{target}分别作为 GRU 算法模型的输入矩阵和目标矩阵。步骤 3-步骤 10：进行 GRU 网络模型的参数更新操作。当训练结束后，使用 GRU 算法模型输出预测联合流量矩阵，作为算法 4-1 中UTM的组成部分。

算法 4-2 网络流量状态预测算法

输入:
学习率: lr
采样大小: $batch$
输入层维度: $input\_dim$
隐藏层维度: $hidden\_dim$
输出层维度: $output\_dim$
时序周期: seq
目标网络更新频率: $freq$
训练总数: $episodes$

联合流量矩阵: UTM

输出:

预测联合流量矩阵

初始化 GRU 网络权重θ

将联合流量矩阵进行时序化操作，得到两个时序化联合流量矩阵

时序化输入联合流量矩阵：$UTM_{input} \leftarrow UTM$

时序化目标联合流量矩阵：$UTM_{target} \leftarrow UTM$

For $episodes$ ←1 to n do:

  初始化$hidden$

  For t ←0 to len(UTM_{input}) do:

    $UTM_{output}^{t+seq+1}, hidden_{output} \leftarrow GRU(UTM_{output}^{t,t+seq}, hidden)$

    使用$loss$=$(UTM_{output}^{t+seq+1} - UTM_{target}^{t+seq+1})^2$执行梯度下降更新 GRU 网络权重参数$\theta$

    $hidden \leftarrow hidden_{output}$

  End for

End for

§4.3.3 MDRL-TP 多智能体跨域路由算法

算法 4-3 是基于本文第三章中所提出的 DRL-TP 算法的改进形式，根据发送流bw的类型用于生成本地控制器中最优域内路由转发路径和根控制器中最优域间路由转发路径。MDRL-TP 多智能体跨域路由算法主要由部署在多个智能体上改进的 DRL-TP 智能路由算法组成，如算法 4-4 所示。步骤 2：表示每次发送网络数据流的大小。步骤 3-步骤 6：分别获取三个网络子域中的最优域内路由转发路径$intra_optimal_path_{D_1}$、$intra\_optimal\_path_{D_2}$、$intra\_optimal\_path_{D_3}$。步骤 7：根控制器通过协同通信模块中 socket 技术主动间隔式获取全局网络中的联合流量矩阵UTM。步骤 8：根据每个网络子域中的域间路由转发请求生成最优域间路由转发路径$inter_optimal_path$。步骤 9：根据已获取的每个网络子域中最优域内路由转发路径和网络子域间的最优域间路由转发路径，经过 MDRL-TP 多智能体跨域路由算法生成全局网络中所有源-目的交换机节点的最优路由转发路径$all_optimal_path$。

算法 4-3 改进的 DRL-TP 智能路由算法

输入:

Dueling DQN 算法模型

GRU 预测算法模型

发流大小：bw_list

输出:

每个控制器中所有源-目的交换机节点的最优转发路径

加载 Dueling DQN 及 GRU 预测算法模型，构建改进的 DRL-TP 算法模型

For $bw \leftarrow$ 1 to bw_list do:
  If bw为本地控制器中的发送流:
    $intra\_optimal\_path \leftarrow$ DRL-TP(TM)
  Else if bw 在根控制器中的发送流:
    $inter\_optimal\_path \leftarrow$ DRL-TP(UTM)
  End if
End for

算法 4-4 MDRL-TP 多智能体路由算法

输入:

改进的 DRL-TP 智能路由算法模型

发送流大小: bw_list

输出:

整个网络中所有源-目的交换机节点的最优转发路径

加载改进的 DRL-TP 路由算法模型，构建 MDRL-TP 多智能体路由算法模型
For $bw \leftarrow$ 1 to bw_list do:
  For $D_d \leftarrow$ **1 to 3 do:**
    获取当前时刻网络子域下的流量矩阵TM_{D_d}
    $intra\_optimal\_path_{D_i} \leftarrow$ DRL-TP(TM_{D_d})
  End for
  根控制器通过协同通信模块获取联合流量矩阵UTM
  根控制器生成最优域间路由转发路径 $inter\_optimal\_path$ ←DRL-TP(UTM)
  $all\_optimal\_path$ ←MDRL-TP($intra\_optimal\_path_{D_1 \cup D_2 \cup D_3} \cup inter\_optimal\_path$)
End for

§4.4 实验与结果分析

§4.4.1 实验环境与测试

本章使用了四台 Ubuntu16.04 内存大小为 2G，4 核处理器的系统用于搭建 SDN 多控制器分层式架构，其中三台作为本地控制器，一台作为根控制器。四台 Ubuntu16.04 系统都安装了 Mininet 2.3.0 和 Ryu4.34 用于搭建 SDN 网络环境，其中 SDN 网络环境中的拓扑由 Mininet 生成，SDN 控制器使用基于 Python 开源的 Ryu，并使用 Iperf 工具用于创建和发送网络中的数据流。如图 4-6 所示，网络拓扑总共包含 39 个节点，为修改后的纽约市中心共划分为三个网络子域，每个节点代表一台支持 OpenFlow1.3 协议的交换机，每台交换机下挂载一台主机。考虑到实验硬件设备的限制以及新型异构网络的需求，交换机之间的传输链路带宽大小在[1Mbit-10Mbit]内随机设定。

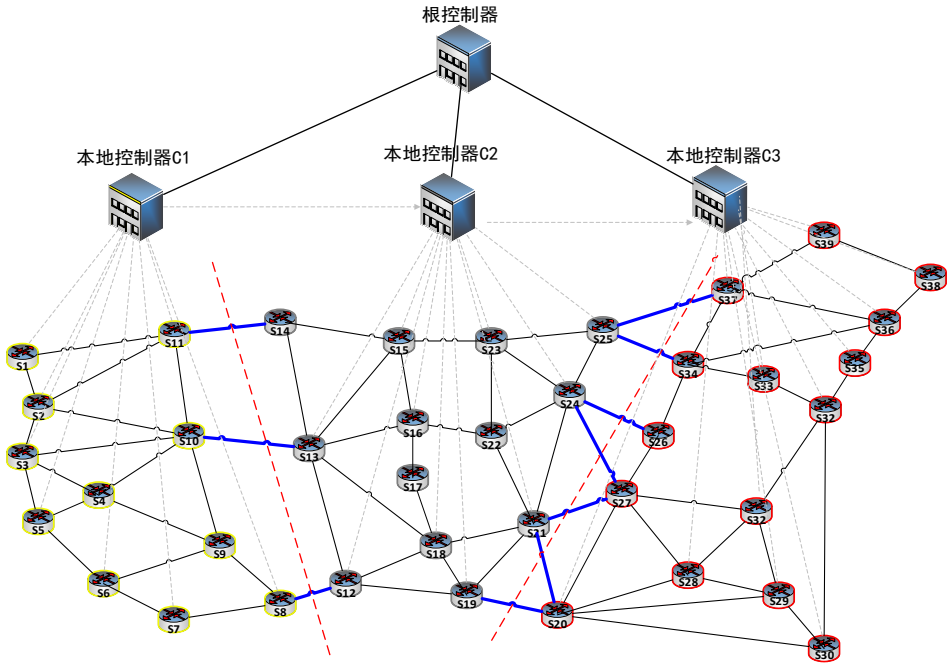
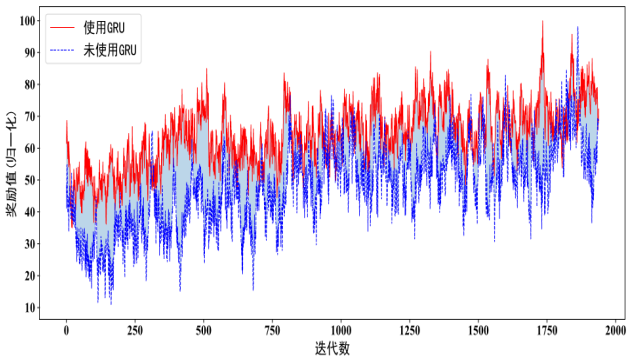


图 4-6 修改后 39 个节点的纽约市中心网络拓扑图

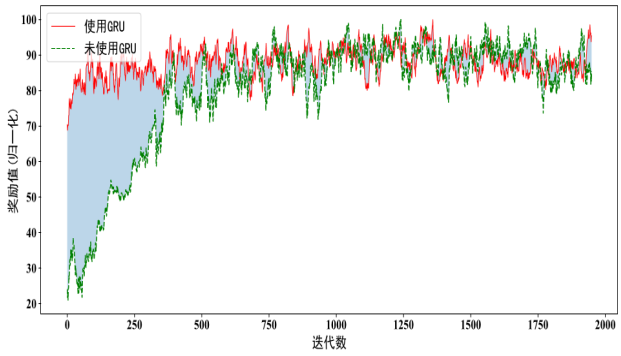
§4.4.2 预测算法性能与实验参数分析

从图 4-7 可以看出，使用 GRU 网络流量状态预测算法能够使智能体获得更高的奖励值。这是由于 GRU 预测算法一方面能够发现 SDN 多控制器管理下的大规模网络中隐藏和未知的网络流量状态，而基于 SDN 多线程网络子域测量机制和协同通信模块一般很难获取这些隐藏和未知的网络流量状态；另一方面 GRU 预测算法能够预测网络流量状态的未来变化趋势，使得 MDRL-TP 多智能体跨域路由算法探索到更大的状态空间，学习到更优的动作策略，从而获得更高的奖励，验证了使用 GRU 网络流量状态预测算法能够在一定程度上提升 MDRL-TP 多智能体跨域路由算法的性能。在本文第三章基于 SDN 单控制器管理下的智能路由优化算法中，对组成流量矩阵和奖励函数的最优权重因子以及组成流量矩阵的网络链路指标个数，做了详细的参数对比实验。考虑到联合流量矩阵数据分布的相似性，本章沿用 DRL-TP 智能路由算法的设计思路，使用$[0.6,0.3,0.1,0.1,0.1,0.1]$和$[0.5,-0.4,-0.3,-0.3,-0.3,-0.3]$作为组成$UTM$的和$UR$的权重因子，并使用带宽、时延、丢包率、已用带宽、弃包数和错包数六个网络链路指标组成UTM。

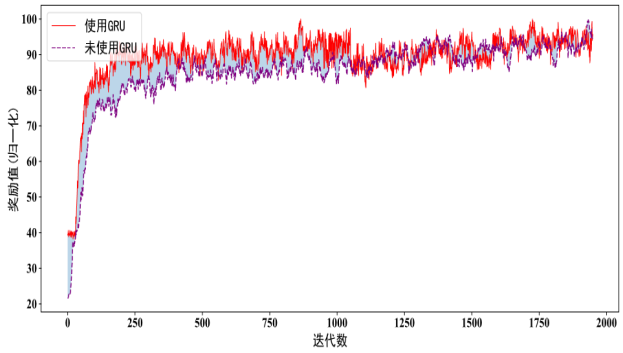
从图 4-8 和图 4-9 可以得出一些结论：（1）相较于本地控制器中的智能体 1、2 和 3，根控制器下的智能体中 GRU 算法模型损失值以及 DRL 算法模型的奖励收敛迭代数要大，这是因为根控制器中的联合状态空间由UTM组成，而联合流量矩阵的维度要远高于三个本地控制器中的域内流量矩阵TM的维度；（2）智能体 1 获取的奖励值比



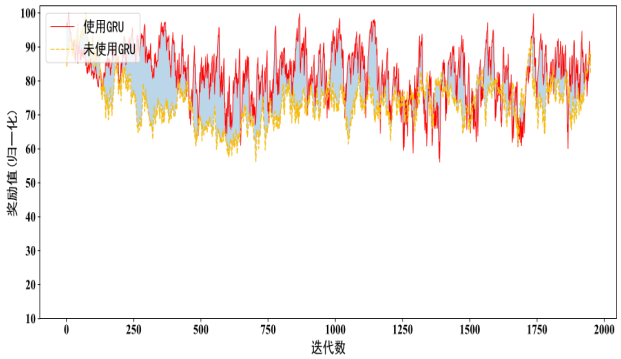
（a）根控制器中的智能体



（b）本地控制器 C1 中的智能体 1



（c）本地控制器 C2 中的智能体 2



（d）本地控制器 C3 中的智能体 3

图 4-7 使用 GRU 与未使用 GRU 跨域路由算法奖励回报对比图

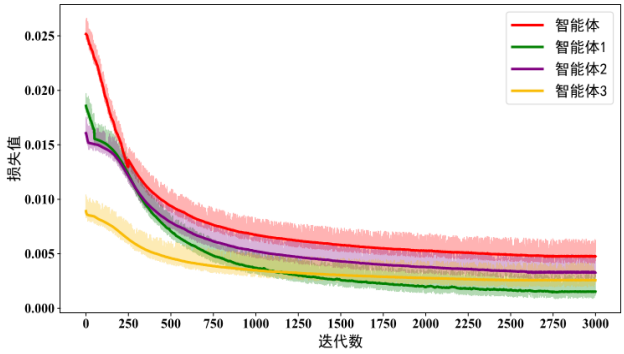
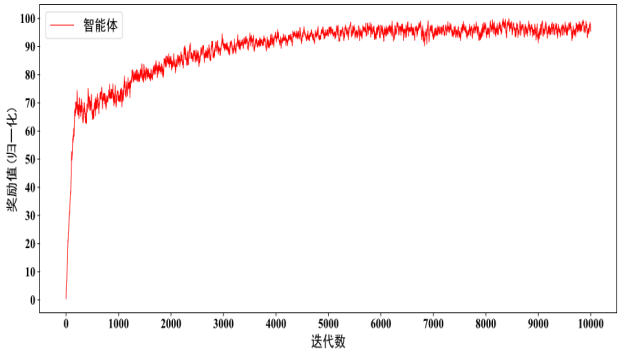
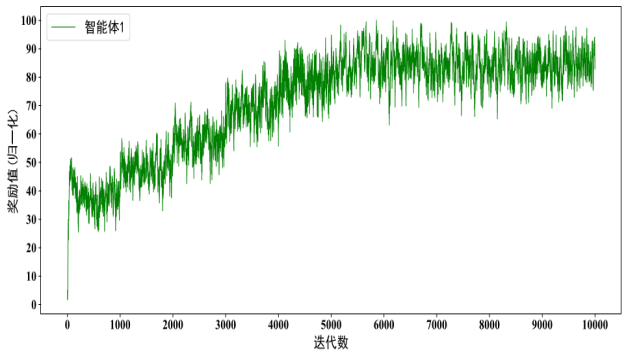


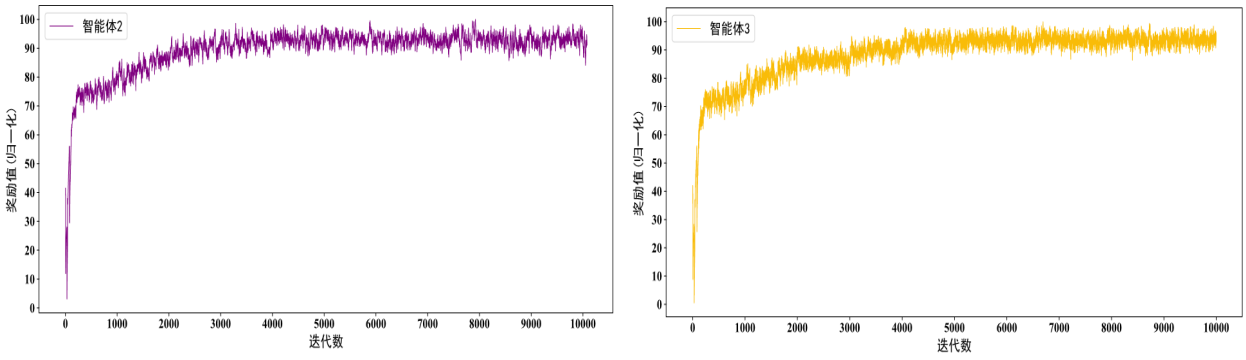
图 4-8 网络流量状态预测算法损失曲线



（a）



（b）



（c）　　（d）

图 4-9 奖励回报曲线图

其他智能体获取的奖励值低，收敛程度也比其他智能体差，主要原因是为了验证在某个网络子域处于拥塞时，本章提出的 MDRL-TP 多智能体跨域路由算法能否保证全局网络的性能。因此，将 C1 管理下的网络子域中的链路带宽大多数限定在 5Mbit 以内。随着发送网络数据流的增大，C1 管理下的网络子域更容易处于拥塞状态，导致获取的域内流量矩阵性能较差，从而影响 DRL 算法模型的收敛程度。

§4.4.3 对比实验及结果

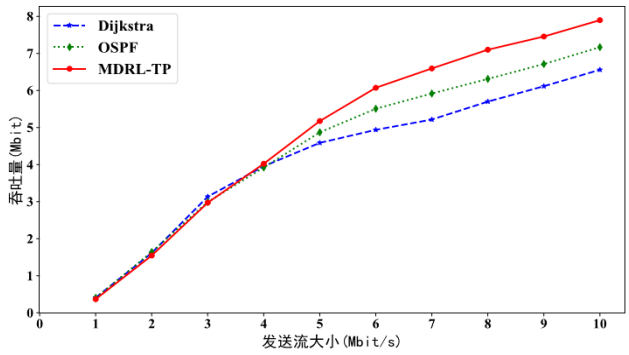
本章使用迪杰斯特拉（Dijkstra）和开放最短路径优先（OSPF）两种路由算法与 MDRL-TP 多智能体跨域路由算法进行网络性能对比，下面分别介绍 Dijkstra 和 OSPF 两种路由算法的设计思路。

Dijkstra 路由算法：首先，在构建 SDN 网络拓扑时，将整个网络拓扑中所有交换机节点之间的链路权重值W设置为 1。然后，基于跳数最短作为每个源-目的交换机节点的最优路由转发路径。

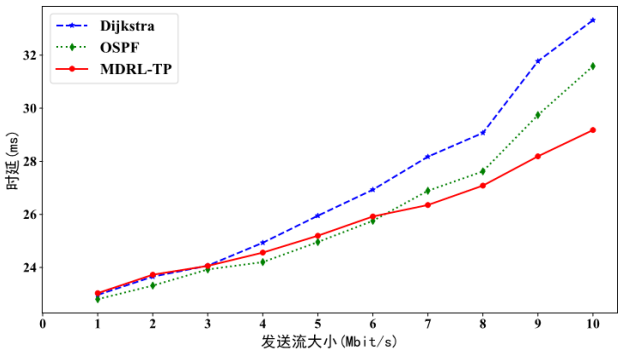
OSPF 路由算法：首先，通过 SDN 多线程网络子域测量机制用于获取每个子域中域内链路时延，通过发送探测包$detect_pkt$获取网络子域间的域间链路时延，进而获取全局网络中的链路时延。然后，根据全局网络链路时延得到所有源-目的交换机节点之间的转发路径。最后，从中选择跳数最短的路径作为所有源-目的交换机节点的最优路由转发路径。

本章使用第二章中基于 SDN 多控制器下所设计的网络吞吐量、网络时延和网络丢包率三个性能指标用于评估三种路由算法对网络性能的影响。为了更好的验证本章 MDRL-TP 跨域智能路由算法的有效性，分别对每个网络子域中的域内网络指标和全局网络中的网络指标进行实验对比。首先，对每个网络子域中域内网络性能指标的对比。如图 4-10、图 4-11 和图 4-12 所示，可以得出一些结论：（1）相较于本地控制器 C2 与 C3 管理下的网络子域中，本地控制器 C1 管理下的网络子域中域内平均吞吐量

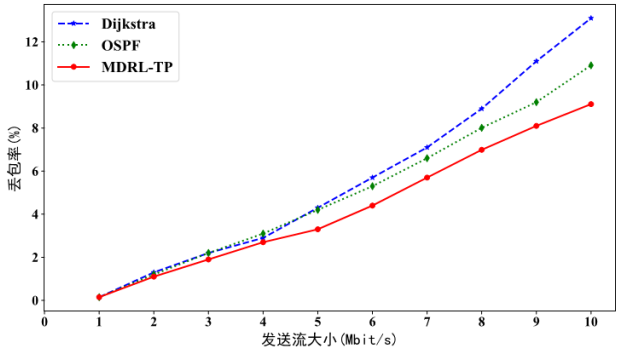
明显较低。这是由于在本章实验中，整个网络拓扑的链路带宽在[1Mbit-10Mbit]内随机设置，而为了更好验证网络处于拥塞状态下本章提出的 MDRL-TP 多智能体跨域路由算法的有效性，将 C1 管理下的网络子域中的链路带宽大多数限定在 5Mbit 以内。因此，当发送流大小超过 5Mbit/s 后，C1 管理下的网络子域中出现了较为明显的拥塞现象，导致图 4-10（b）中的域内平均时延和图 4-10（c）中的域内平均丢包率迅速增大，从而影响了域内平均吞吐量的大小；（2）本地控制器 C2 和 C3 管理下的网络子域中的域内平均吞吐量、时延和丢包率相差不大，但是 C3 管理下的网络子域中的域内网络性能指标要优于 C2，一方面是由于在随机设置网络链路带宽大小时，C3 管理下的网络链路带宽整体上大于 C2 管理下的网络链路带宽，另一方面是由于本地控制



（a）域内平均吞吐量

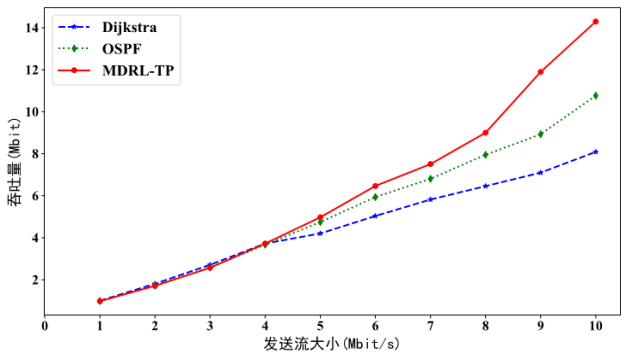


（b）域内平均时延

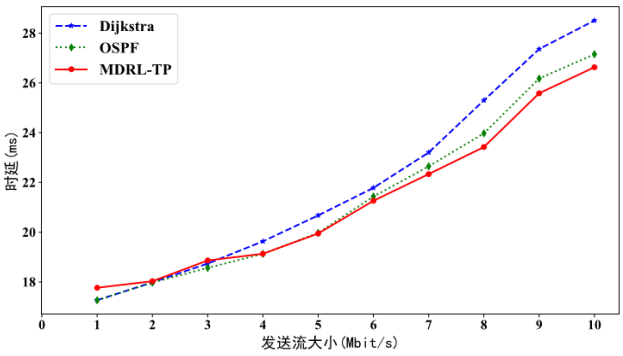


（c）域内平均丢包率

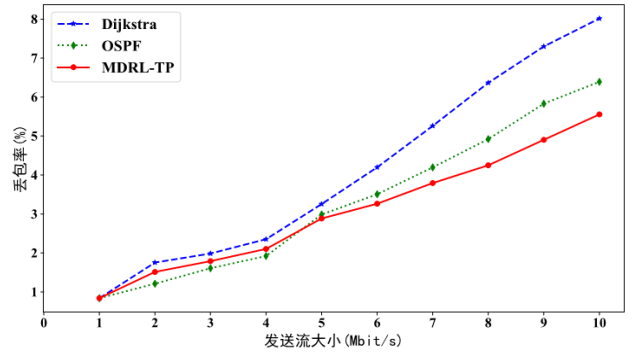
图 4-10 本地控制器 C1 域内网络性能指标对比



（a）域内平均吞吐量

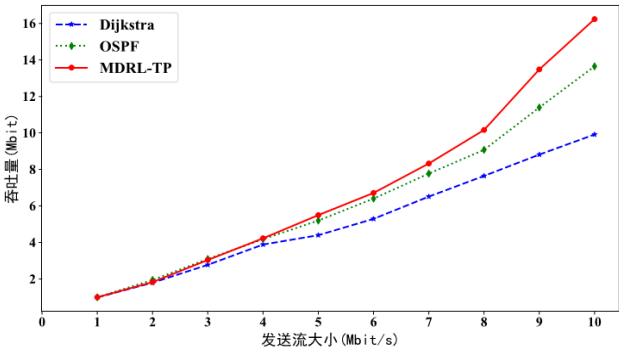


（b）域内平均时延

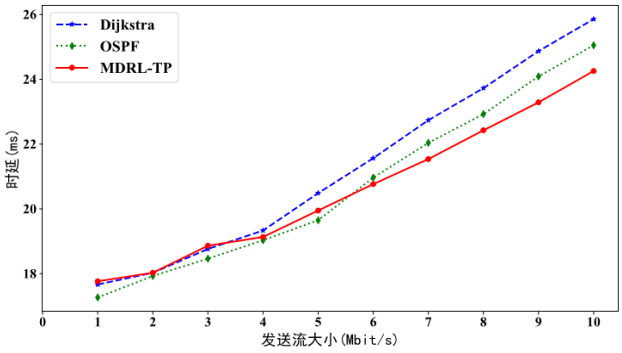


（c）域内平均丢包率

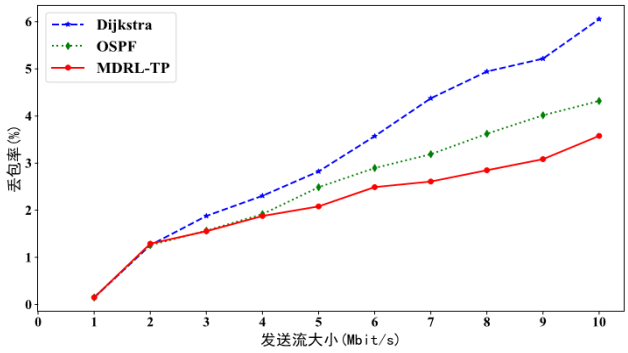
图 4-11 本地控制器 C2 域内网络性能指标对比



（a）域内平均吞吐量



（b）域内平均时延



（c）域内平均丢包率

图 4-12 本地控制器 C3 域内网络性能指标对比

器 C2 作为中间控制器，在进行域间信息交换时，当本地控制器 C1 处于拥塞状态时，对其产生的影响大于C3。因此，C3 管理下的网络子域的域内网络性能指标更加优异；（3）当网络处于正常或拥塞状态时，相较于 Dijkstra 和 OSPF 两种路由算法，本章提出的 MDRL-TP 多智能体跨域路由算法能够使每个网络子域中域内网络性能指标表现更加优异，验证了多智能体跨域路由算法的有效性。

接下来，进行全局网络的性能指标对比。从图 4-13、图 4-14 和图 4-15 可以得出一些结论：（1）当发送流大小达到 5Mbit/s 时，全局网络中的平均吞吐量、时延和丢包率有较大程度变化。因此，可以断定网络出现了一定程度的拥塞现象，而相较于

Dijkstra 和 OSPF 两种路由算法，MDRL-TP 多智能体跨域路由算法仍然能够保证全局网络的性能不受太大影响；（2）全局网络的性能会受到某个网络子域的性能影响，因此，相较于图 4-11（a）和图 4-12（a）而言，图 4-13 所示的网络平均吞吐量较低，这是由于本地控制器 C1 管理下的网络子域中，由于所设定的链路带宽大小的影响，在发送流大于 5Mbit/s 时该网络子域出现了严重拥塞现象，不仅影响了本网络子域的性能，而且对全局网络的性能造成了一定的影响；（3）即使某个网络子域中的网络性能较差时，本章提出的 MDRL-TP 多智能体跨域路由算法仍然能够保证全局网络的性能处于相对良好的状态，验证了 MDRL-TP 多智能体跨域路由算法的有效性。

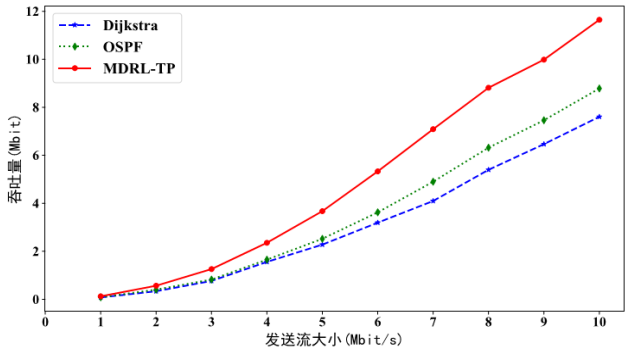


图 4-13 全局网络平均吞吐量

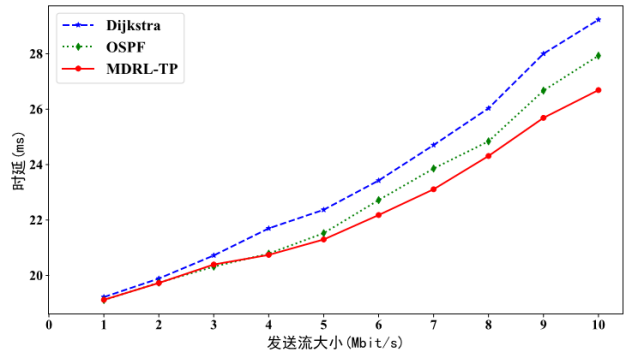


图 4-14 全局网络平均时延

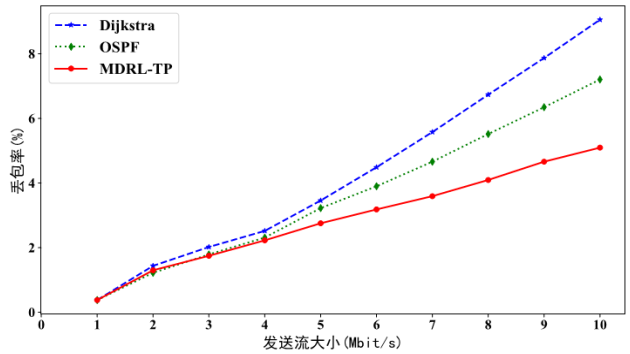


图 4-15 全局网络平均丢包率

§4.5 本章小结

随着 SDN 网络规模不断增大，基于 SDN 单控制器的智能路由算法存在控制器负载过大、控制器单点故障和网络数据包堆积等问题。为了解决大规模 SDN 路由优化问题，一种可行的方法是将大规模网络划分成多个网络子域并使用多个 SDN 控制器进行管理，基于此本章提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法，每个网络子域中的最优域内路由转发路径由本地控制器生成，而最优域间路由转发路径由每个本地控制器通过协同通信模块中的 socket 技术请求根控制器生成，最后生成全局网络的最优路由转发路径。与 Dijkstra 和 OSPF 两种路由算法相比，MDRL-TP 多智能体跨域路由算法不仅能够保证每个网络子域中的域内网络性能，而且使得全局网络中的平均吞吐量、时延和丢包率都有显著提升，对解决大规模 SDN 网络路由优化问题具有一定的实用价值。

第五章 总结与展望

§5.1 论文工作总结

近年来，随着云计算、大数据和 5G 网络等新技术的出现，导致各种类型的多媒体服务和网络设备的快速发展，这些网络设备通常具有响应时间快、网络性能高、功能类型多等特点，以满足当前用户对 QoS 的需求。其中，网络性能质量是满足 QoS 需求最基本且最重要的条件，这就需要一个高效的 QoS 感知网络架构来保证网络性能的质量。SDN 作为一种新的网络架构，通过将数据与控制平面解耦，并对网络进行集中统一管理，能够很方便获取全局网络视图，达到灵活部署路由的策略，从而提高网络性能。因此，SDN 被认为是一种十分有效的 QoS 感知网络架构。传统的路由方法仅使用有限的网络状态信息进行路由决策，且存在适应动态复杂网络变化能力差、调整路由转发策略不灵活的缺陷。因此，无法实现自适应的智能化路由决策。随着人工智能技术的快速发展，尤其是深度强化学习算法在处理高维序列化决策问题上展现出的巨大优势，已经被广泛应用于路由优化问题的研究。因此，针对传统路由方法存在的缺陷，本文通过设计一种基于深度强化学习机制下的 SDN 智能路由优化算法，实现了自适应的路由决策，有效的提升了网络的性能。同时，随着网络规模的不断增大，网络中将会存在高量的数据流量，针对 SDN 单控制器管理下的大规模网络中存在控制器负载过大等问题，致使网络数据包产生堆积现象，导致无法实时进行智能化路由决策，本文通过设计一种基于 SDN 多智能体的跨域路由算法，将大规模网络划分成多个控制器进行分域管理，由本地控制器负责处理本域中的网络数据流并生成最优域内路由决策，根控制器负责响应域间请求并生成最优域间路由决策，实现了大规模网络中实时智能化最优路由决策。本文主要工作总结如下：

（1）针对传统路由方法仅使用有限网络链路信息进行路由决策，且适应动态复杂网络变化能力差、调整路由转发策略不灵活的缺陷，提出了一种基于 Dueling DQN 强化学习和网络流量状态预测的 SDN 智能路由方法，通过获取全局网络链路状态信息，实时自适应生成最佳路由转发决策，有效的提升了基于 SDN 单控制器管理下的网络性能。

（2）针对大规模网络中 SDN 单控制器存在负载过大，导致出现网络数据包堆积现象而无法进行实时路由转发操作，加上传统域间路由方法存在配置繁琐和获取网络状态信息不够灵活等缺陷，导致难以获取全局网络状态信息，从而不能自适应生成最佳路由决策，提出了一种基于多智能体深度强化学习和网络流量状态预测的 SDN 跨域智能路由方法，实现了大规模网络中实时智能化最优路由决策的能力，有效的提升

了 SDN 多控制器管理下的网络性能。

§5.2 未来展望

本文接下来的研究工作可以从以下几个方面展开：

（1）本文使用 GRU 作为网络流量状态预测算法，下一步可以尝试多层 GRU 结构或者结合图神经网络的预测算法，以更好的预测网络流量状态的未来变化趋势，提升智能路由优化算法的感知能力。

（2）为了增大 socket 通信时每次传输的数据大小，减少根-本地控制器之间的交互次数，本文使用缓冲区技术用于临时存储当前已获取网络子域中的状态信息，在满足一定条件时，才将缓冲区中的状态信息批次传输到根控制器中，从而获取全局网络状态信息。因此，缓冲区大小的设置是否会影响 MDRL-TP 多智能体跨域路由算法的性能，将是下一步研究的方向。

（3）SDN 多控制器部署的位置以及划分网络子域的数量都会对多智能体跨域路由优化算法产生影响。因此，如何确定多控制器部署位置以及网络子域的数量，是值得进一步研究的工作。

参考文献

[1] McKeown N. Software-defined networking[J]. INFOCOM keynote talk, 2009, 17(2): 30-32.

[2] Sun P, Yu M, Freedman M J, et al. HONE: Joint Host-Network Traffic Management in Software-Defined Networks[J]. Journal of Network and Systems Management, 2015, 23(2):374-399.

[3] Guo Z, Dou S, Wang Y, et al. HybridFlow: Achieving load balancing in software-defined WANs with scalable routing[J]. IEEE Transactions on Communications, 2021, 69(8): 5255-5268.

[4] Tulloh R, Ginting J G A, Mulyana A, et al. Performance Comparison of File Transfer Protocol Service Between Link State and Distance Vector Routing Protocol in Software Defined Network, In Proceedings of the IOP Conference Series: Materials Science and Engineering, September 1-2, 2020[C]. IOP Publishing

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

[6] Ahn C W, Ramakrishna R S. A genetic algorithm for shortest path routing problem and the sizing of populations[J]. IEEE transactions on evolutionary computation, 2002, 6(6): 566-579.

[7] Derbel H, Jarboui B, Hanafi S, et al. Genetic algorithm with iterated local search for solving a location-routing problem[J]. Expert Systems with Applications, 2012, 39(3): 2865-2871.

[8] Zhang D, Liu S, Liu X, et al. Novel dynamic source routing protocol (DSR) based on genetic algorithm‐bacterial foraging optimization (GA‐BFO)[J]. International Journal of Communication Systems, 2018, 31(18): 1-20.

[9] Parsaei M R, Mohammadi R, Javidan R. A new adaptive traffic engineering method for telesurgery using ACO algorithm over software defined networks[J]. European Research in Telemedicine/La Recherche Europeenne en Telemedecine, 2017, 6(3-4): 173-180.

[10] Jing S, Muqing W, Yong B, et al. An improved GAC routing algorithm based on SDN, In Proceedings of the 3rd IEEE International Conference on Computer and Communications (ICCC), April 27-30, 2017[C]. Nagoya, Japan: IEEE.

[11] Lin C, Wang K, Deng G. A QoS-aware routing in SDN hybrid networks[J]. Procedia Computer Science, 2017, 110: 242-249.

[12] Truong Dinh K, Kukliński S, Osiński T, et al. Heuristic traffic engineering for SDN[J]. Journal of Information and Telecommunication, 2020, 4(3): 251-266.

[13] Ke C K, Wu M Y, Hsu W H, et al. Discover the Optimal IoT Packets Routing Path of Software-Defined Network via Artificial Bee Colony Algorithm, In Proceedings of the Wireless Internet: 12th EAI International Conference, November 26-27, 2019[C]. Springer.

[14] Shokouhifar M. FH-ACO: Fuzzy heuristic-based ant colony optimization for joint virtual network function placement and routing[J]. Applied Soft Computing, 2021, 107(2): 107401.

[15] Zhang L, Lei Y. Particle swarm optimization‐based information‐centric networking intra‐domain routing strategy[J]. Internet Technology Letters, 2021, 4(1): e196.

[16] Valadarsky A, Schapira M, Shahaf D, et al. Learning to route, In Proceedings of the 16th ACM workshop on hot topics in networks, November 30-December 1, 2017[C]. Palo Alto, CA, USA: ACM.

[17] Sharma D K, Dhurandher S K, Woungang I, et al. A machine learning-based protocol for efficient routing in opportunistic networks[J]. IEEE Systems Journal, 2016, 12(3): 2207-2213.

[18] Li W, Li G, Yu X. A fast traffic classification method based on SDN network, In Proceedings of the 4th International Conference on Electronics, July 28-31, 2014[C]. Beijing, China.

[19] Zhou X, Su M, Liu Z, et al. Smart Tour Route Planning Algorithm Based on Naïve Bayes Interest Data Mining Machine Learning[J]. ISPRS International Journal of Geo-Information, 2020, 9(2): 112.

[20] Yanjun L, Xiaobo bL, Osamu Y. Traffic engineering framework with machine learning based meta-layer in software-defined networks, In Proceedings of the 4th IEEE International Conference on Network Infrastructure and Digital Content, September 19-21, 2014[C]. Beijing, China: IEEE.

[21] Tang F, Mao B, Fadlullah Z M, et al. On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control[J]. IEEE Wireless Communications, 2017, 25(1): 154-160.

[22] Mao B, Tang F, Fadlullah Z M, et al. An intelligent route computation approach based on real-time deep learning strategy for software defined communication systems[J]. IEEE Transactions on Emerging Topics in Computing, 2019, 9(3): 1554-1565.

[23] Kato N, Fadlullah Z M, Mao B, et al. The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective[J]. IEEE wireless communications, 2016, 24(3): 146-153.

[24] Hendriks T, Camelo M, Latré S. Q 2-routing: A Qos-aware Q-routing algorithm for wireless ad hoc networks, In Proceedings of the 14th International Conference on Wireless and Mobile Computing, June 25-29, 2018[C]. Chongqing, China: IEEE.

[25] Chen T, Gao X, Liao T, et al. Pache: A Packet Management Scheme of Cache in Data Center Networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2019, 31(2): 253-265.

[26] Casas-Velasco D M, Rendon O M C, da Fonseca N L S. Intelligent routing based on reinforcement learning for software-defined networking[J]. IEEE Transactions on Network and Service Management, 2020, 18(1): 870-881.

[27] Jin Z, Zang W, Jiang Y, et al. A QLearning Based Business Differentiating Routing Mechanism in

SDN Architecture, In Proceedings of the Journal of Physics: Conference Series, February 22-24, 2019[C]. IOP Publishing.

[28] Yin Y, Huang C, Wu D F, et al. Reinforcement Learning-Based Routing Algorithm in Satellite-Terrestrial Integrated Networks[J]. Wireless Communications and Mobile Computing, 2021, 2021.

[29] Zhao L, Wang J, Liu J, et al. Routing for crowd management in smart cities: A deep reinforcement learning perspective[J]. IEEE Communications Magazine, 2019, 57(4): 88-93.

[30] 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.

[31] Zhang J, Ye M, Guo Z, et al. CFR-RL: Traffic engineering with reinforcement learning in SDN[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(10): 2249-2259.

[32] Fu Q, Sun E, Meng K, et al. Deep Q-learning for routing schemes in SDN-based data center networks[J]. IEEE Access, 2020, 8: 103491-103499.

[33] Liu W, Cai J, Chen Q C, et al. DRL-R: Deep reinforcement learning approach for intelligent routing in software-defined data-center networks[J]. Journal of Network and Computer Applications, 2021, 177: 102865.

[34] Hossain M B, Wei J. Reinforcement learning-driven QoS-aware intelligent routing for software-defined networks, In Proceedings of the 2019 IEEE global conference on signal and information processing (GlobalSIP), November 11-14, 2019[C]. USA: IEEE.

[35] Yu C, Lan J, Guo Z, et al. DROM: Optimizing the routing in soft-ware-defined networks with deep reinforcement learning[J]. IEEE Access, 2018, 6: 64533-64539.

[36] Wang H, Xu H, Huang L, et al. Load-balancing routing in software defined networks with multiple controllers[J]. Computer Networks, 2018, 141: 82-91.

[37] EL-Garoui L, Pierre S, Chamberland S. A new SDN-based routing protocol for improving delay in smart city environments[J]. Smart Cities, 2020, 3(3): 1004-1021.

[38] Moufakir T, Zhani M F, Gherbi A, et al. Collaborative multi-domain routing in SDN environments[J]. Journal of Network and Systems Management, 2022, 30(1): 1-23.

[39] Xu A, Sun S, Wang Z, et al. Multi-Controller Load Balancing Mechanism Based on Improved Genetic Algorithm, In Proceedings of the 2022 International Conference on Computer Communications and Networks (ICCCN), July 25-28, 2022[C]. IEEE.

[40] Zhang B, Wang X, Huang M. Adaptive consistency strategy of multiple controllers in SDN[J]. IEEE Access, 2018, 6: 78640-78649.

[41] Podili P, Kataoka K. TRAQR: Trust aware End-to-End QoS routing in multi-domain SDN using Blockchain[J]. Journal of Network and Computer Applications, 2021, 182: 103055.

[42] Li Z, Zhou X, Gao J, et al. SDN controller load balancing based on reinforcement learning, In

Proceedings of the 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), November 23-25, 2018[C]. Beijing, China: IEEE.

[43] Casas-Velasco D M, Rendon O M C, da Fonseca N L S. DRSIR: A Deep Reinforcement Learning Approach for Routing in Software-Defined Networking[J]. IEEE Transactions on Network and Service Management, 2021.

[44] Wang Y C, Chang E J. Cooperative flow management in multi-domain SDN-based networks with multiple controllers, In Proceedings of the 2020 IEEE 17th International Conference on Smart Communities: Improving Quality of Life Using ICT, April 4-7, 2020[C]. Iowa, USA: IEEE.

[45] Godfrey D, Jang J, Kim K I. Weight Adjustment Scheme Based on Hop Count in Q-routing for Software Defined Networks-enabled Wireless Sensor Networks[J]. Journal of information and communication convergence engineering, 2022, 20(1): 22-30.

[46] Yuan T, da Rocha Neto W, Rothenberg C E, et al. Dynamic controller assignment in software defined internet of vehicles through multi-agent deep reinforcement learning[J]. IEEE Transactions on Network and Service Management, 2020, 18(1): 585-596.

[47] Sun P, Guo Z, Wang G, et al. MARVEL: Enabling controller load balancing in software-defined networks with multi-agent reinforcement learning[J]. Computer Networks, 2020, 177: 107230.

[48] Mazyavkina N, Sviridov S, Ivanov S, et al. Reinforcement learning for combinatorial optimization: A survey[J]. Computers & Operations Research, 2021, 134: 105400.

[49] Nian R, Liu J, Huang B. A review on reinforcement learning: Introduction and applications in industrial process control[J]. Computers & Chemical Engineering, 2020, 139: 106886.

[50] Agarwal A, Kakade S M, Lee J D, et al. Optimality and approximation with policy gradient methods in markov decision processes, In Proceedings of the Conference on Learning Theory, July 9-12, 2020[C]. Graz, Austria: PMLR.

[51] Jang B, Kim M, Harerimana G, et al. Q-learning algorithms: A comprehensive classification and applications[J]. IEEE Access, 2019, 7: 133653-133667.

[52] Wang Y H, Li T H S, Lin C J. Backward Q-learning: The combination of Sarsa algorithm and Q-learning[J]. Engineering Applications of Artificial Intelligence, 2013, 26(9): 2184-2193.

[53] Nguyen T T, Nguyen N D, Nahavandi S. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications [J]. IEEE transactions on cybernetics, 2020, 50(9): 3826-3839.

[54] LeCun Y, Bengio Y, Hinton G. Deep learning [J]. Nature, 2015, 521(7553): 436-444.

[55] Sewak M, Sewak M. Deep Q Network (DQN), Double DQN, and Dueling DQN: A Step Towards General Artificial Intelligence[J]. Deep Reinforcement Learning: Frontiers of Artificial Intelligence, 2019: 95-108.

[56] Meng L, Yang W, Guo B, et al. Deep Reinforcement Learning Enabled Network Routing Optimization Approach with an Enhanced DDPG Algorithm, In Proceedings of the Asia Communications and Photonics Conference, October 24-27, 2020[C]. Beijing, China: OSA.

[57] Gu Y, Cheng Y, Chen C L P, et al. Proximal policy optimization with policy feedback[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2021, 52(7): 4600-4610.

[58] Hessel M, Modayil J, Hasselt H, et al. Rainbow: Combining improvements in deep reinforcement learning, In Proceedings of the AAAI conference on artificial intelligence, February 2-7, 2018[C]. New Orleans, USA: AAAI.

[59] Liu M, Zhang H, Chen Y, et al. An adaptive timing mechanism for urban traffic pre-signal based on hybrid exploration strategy to improve double deep Q network[J]. Complex & Intelligent Systems, 2023, 9(2): 2129-2145.

[60] Schulman J, Levine S, Abbeel P, et al. Trust region policy optimization, In Proceedings of the International conference on machine learning, July 6-11 2015[C]. Guangzhou, China: PMLR.

[61] Yu Y, Si X, Hu C, et al. A review of recurrent neural networks: LSTM cells and network architectures[J]. Neural Computation, 2019, 31(7): 1235-1270.

[62] Van Houdt G, Mosquera C, Nápoles G. A review on the long short-term memory model[J]. Artificial Intelligence Review, 2020, 53(2): 5929-5955.

[63] Fan J, Mu D, Liu Y. Research on network traffic prediction model based on neural network, In Proceedings of the 2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE), September 23-25, 2019[C]. Vancouver, Canada: IEEE.

[64] Yang L, Cai Z P, Xu H. LLMP: exploiting LLDP for latency measurement in software-defined data center networks[J]. Journal of Computer Science and Technology, 2018, 33(2): 277-285.

[65] 兰巨龙,张学帅,胡宇翔,孙鹏浩.基于深度强化学习的软件定义网络 QoS 优化[J].通信学报,2019,40(12):60-67.

[66] Clark D D, Partridge C, Ramming J C, et al. A knowledge plane for the internet, In Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, August 23-25, 2003[C]. New York, USA: ACM.

[67] Mestres A, Rodriguez-Natal A, Carner J, et al. Knowledge-defined networking[J]. ACM SIGCOMM Computer Communication Review, 2017, 47(3): 2-10.

[68] Alexandropoulos S A N, Kotsiantis S B, Vrahatis M N. Data preprocessing in predictive data mining[J]. The Knowledge Engineering Review, 2019, 34: e1.

[69] Mininet. Accessed: Jan. 5, 2021. [Online]. Available: http://mininet.org/.

[70] Ryu. Accessed: Dec. 31, 2020. [Online]. Available: https://github.com/faucetsdn/ryu.

[71] iPerf. Accessed: Jan. 5, 2021. [Online]. Available: https://iperf.fr/.

[72] New York Metro IBX data center data sheet. Accessed: Dec. 31, 2020[Online] Available:

https://www.equinix.com/resources/data-sheets/nyc-metro-data-sheet/.

[73] Hu T, Guo Z, Yi P, et al. Multi-controller based software-defined networking: A survey[J]. IEEE Access, 2018, 6: 15980-15996.

[74] Chen C, Liao Z, Ju Y, et al. Hierarchical domain-based multi-controller deployment strategy in SDN-enabled space-air-ground integrated network[J]. IEEE Transactions on Aerospace and Electronic Systems, 2022, 58(6): 4864 - 4879.

[75] Luong N C, Hoang D T, Gong S, et al. Applications of deep reinforcement learning in communications and networking: A survey[J]. IEEE Communications Surveys & Tutorials, 2019, 21(4): 3133-3174.

[76] Xie J, Yu F R, Huang T, et al. A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges[J]. IEEE Communications Surveys & Tutorials, 2018, 21(1): 393-430.

[77] Zhao X, Band S S, Elnaffar S, et al. The implementation of border gateway protocol using software-defined networks: A systematic literature review[J]. IEEE Access, 2021.

[78] Liatifis A, Sarigiannidis P, Argyriou V, et al. Advancing SDN: from OpenFlow to P4, a Survey[J]. ACM Computing Surveys (CSUR), 2022.

[79] Moatemri M, Eltaief H, Kamel A E, et al. Secure East-West Communication to Approve Service Access Continuity for Mobile Devices in a Distributed SDN, In Proceedings of the International Conference on Computational Science and Its Applications, July 4-7, 2022[C]. Dubai, UAE: Springer.

[80] Almadani B, Beg A, Mahmoud A. DSF: A distributed sdn control plane framework for the east/west interface[J]. IEEE Access, 2021, 9(4): 26735-26754.

[81] Yu H, Qi H, Li K. WECAN: An Efficient west-east control associated network for large-scale SDN systems[J]. Mobile Networks and Applications, 2020, 25(1): 114-124.

[82] Oroojlooy A, Hajinezhad D. A review of cooperative multi-agent deep reinforcement learning[J]. Applied Intelligence, 2022, 52 (13): 1-46.

[83] Du W, Ding S. A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications[J]. Artificial Intelligence Review, 2021, 54(5): 3215-3238.

[84] Feriani A, Hossain E. Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial[J]. IEEE Communications Surveys & Tutorials, 2021, 23(2): 1226-1252.

[85] Karamousadakis M, Porichis A, Ottikkutti S, et al. A sensor-based decision support system for transfemoral socket rectification[J]. Sensors, 2021, 21(11): 3743.

致 谢

日月逝矣,岁不我与。在这座丹桂飘香，素有“山水甲天下”美称的国际文化旅游城市，我即将完成研究生的修习。回顾三年的学习生涯，无数校园时光历历在目，心中万千不舍，在这里度过的美好回忆将成为我一生难以忘怀的经历，激我前进，伴我同行。

桃李满天下，春晖遍四方。首先感谢我的导师叶苗教授，他治学严谨、平易近人、兢兢业业、勤奋认真的高尚品格深深教诲着我。感谢叶老师给予读研深造的机会，让我学习了论文撰写、科研创新和学术交流等专业技巧;感谢叶老师给我生活上的帮助，无论是在学校或者在实习，叶老师总是细心询问是否遇到困难，总是倾尽全力的为我着想；感谢叶老师给我工作上的帮助，每次科研中遇到相关瓶颈，您总是及时给予指导并提供宝贵建议，在我因抉择未来的工作而产生迷茫时，您也多次打电话分享自身经验，让我坚定了未来前进的方向。衷心感谢叶老师三年来对我的教导和帮助。

投之以桃,报之以李。其次感谢我的师兄师姐，文鹏、蔡月、魏若愚、陈俊奇、王子辰、戴戬、姜欢、夏亚楠等在学校期间以及毕业后给予我学术和工作上的指导，让我避免了许多弯路。感谢我的同门黄岳劲、张清浩、赵晨玮在科研学习和学科竞赛中的交流和帮助。感谢我实验室的小伙伴张诗杰、陆华成、耿棒棒、李志柯、方兴、肖子涵、曹航竞、周念、李树林、杨思捷等为我的研究生活增添了许多乐趣，很想念和你们一起打气排球的时光。感谢我的师弟师妹胡洪文、李锦强、杨业进、程锦、蒋志邦、王珏、孙石泉、郑基浩、王苏晓，大家和睦相处、风趣诙谐的师门氛围，让我印象深刻。

哀哀父母,生我劬劳。感谢我的父母,他们用弱小的身躯支起了一个幸福的家庭，他们用高尚的品德激励着我奋勇前行，他们平凡的人生却处处彰显着伟大，感谢您们在我读研三年中一直给予我无限的关爱与帮助，希望爸爸妈妈万事吉祥，笑容常伴。感谢我的爷爷奶奶，您们被无情的岁月打磨了青春，但却始终打磨不了对我的疼爱，时常挂念我的吃穿住行,有您们的关爱是我读研最大的动力,希望爷爷奶奶身体健康。感谢我的哥哥，您从小到大一直是我学习的榜样，总是能细心倾听我的诉说并鼓励我直面挫折，不仅是我的人生知己，更是我最敬爱的兄长，希望哥哥工作顺利，事业有成。

最后，感谢所有帮助我，支持我的人。希望自己可以永远保持学习的热情，以积极向上的乐观态度拥抱这个世界。也衷心感谢所有评审专家能够在百忙之中审阅本文，谢谢您们对我的论文提出的宝贵意见！

作者在攻读硕士期间的主要研究成果

论文：

[1] **Huang L**, Ye M, Xue X, et al. Intelligent routing method based on Dueling DQN reinforcement learning and network traffic state prediction in SDN[J]. Wireless Networks, 2022: 1-19，已见刊，SCI JCR3 区（排名第一，导师第二）

[2] Ye M, **Huang L**, Deng X, et al. A New Intelligent Cross-Domain Routing Method in SDN Based on a Proposed Multiagent Reinforcement Learning Algorithm. IEEE Internet of Things Journal[J], 2023，在审，SCI 中科院 1 区（排名第二，导师第一）

科研项目：

[1] 《EB 级集群存储性能优化及节点间高效通信方法研究》, 国家自然科学基金项目, 2019.01-2022.12，参与结题

[2] 《基于 SDN 的工业物联网数据云边协同层级化存储及性能优化方法研究》，国家自然科学基金项目，在研

[3] 《工业物联网中的智能异常检测与智能路由方法研究》，广西研究生教育创新计划项目，小组成员

获奖情况：

[1] 荣获 2022“中国高校计算机大赛-网络技术挑战赛”国赛一等奖（队长）

[2] 荣获 2020 学年研究生学业三等奖学金

[3] 荣获 2021 学年研究生学业一等奖学金

[4] 荣获 2022 学年研究生学业一等奖学金