多模态网络分布式控制平面负载均衡研究

涂化清[1],刘 硕[1],方徐鑫[1],马 博[1],李传煌[1*],朱 俊[2],邹 涛[2]
(1. 浙江工商大学,浙江杭州 310018;2. 之江实验室,浙江杭州 311100)

摘 要: 随着工业互联网、车联网、远程医疗等新型服务的快速发展,多模态网络应运而生. 该架构基于“技术体制与网络环境分离”的设计思想,使多种网络模态能够在同一基础平台上共生共存. 然而,现有研究多集中于多模态网络的环境构建、编译优化与网元设计,缺乏对分布式控制平面负载均衡的系统研究. 部分借鉴SDN(Software-Defined Networking)的交换机迁移与动态重分配机制虽可缓解控制器过载,但需在控制器间频繁同步状态信息,迁移开销大、响应延迟高,难以满足多模态网络的实时性与可扩展性要求. 针对上述问题,本文通过对数据平面流量路由的合理规划,优化多模态网络中控制平面的负载分布,提出一种对多种网络模态的流量路由与多模态网元-控制器分配进行联合优化(Joint optimization of Routing and polymorphic network Element Controller Allocation,JRECA)方法. 该方法将不同模态的控制信息规模差异显式纳入优化框架,综合考虑网元分配、路由选择、控制器处理能力与链路带宽等约束. 对于多模态网络的异构特性,本文提出将不同模态控制信息规模差异纳入控制器负载约束的负载均衡机制,构建了同时实现控制平面负载均衡与数据平面吞吐量最大化的统一模型,实现控制平面负载与数据平面吞吐的协同优化,弥补既有研究中两平面割裂求解的不足,并进一步设计具有严格理论保证的“两步走”算法框架. 首先,设计基于最大负载优先的多模态网元-控制器分配算法,确定多模态网元与控制器之间的匹配关系,并通过近似比证明严格界定了算法性能边界;然后,在动态流量环境下,设计基于原始-对偶方法的在线路由算法,并通过竞争比分析给出在线优化的理论性能下界. 在Fat-Tree和ARPANet两种典型拓扑上的仿真实验表明,本文提出的算法在IPv4、IPv6、工控标识、命名数据标识和身份标识5种网络模态下均取得显著性能提升. 与对比算法相比,本文提出的算法可降低17.56%~20.97%的控制器负载,并提高13.86%~29.82%的系统吞吐量.

关键词: 多模态网络;分布式;控制平面;负载均衡;路由;近似算法
基金项目: 国家自然科学基金(No.U22A2005);浙江省重点研发计划(No.2024SSYS0001)
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2025)11-3996-14
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20250421

Research on Load Balancing of Distributed Control Plane in Polymorphic Network

TU Hua-qing[1], LIU Shuo[1], FANG Xu-xin[1], MA Bo[1], LI Chuan-huang[1*], ZHU Jun[2], ZOU Tao[2]
(1. *Zhejiang Gongshang University*, *Hangzhou*, *Zhejiang* 310018, *China*;
2. *Zhejiang Lab*, *Hangzhou*, *Zhejiang* 311100, *China*)

Abstract: With the rapid development of emerging services such as the Industrial Internet, Internet of Vehicles, and telemedicine, multimodal networks have emerged. This architecture is based on the design principle of “separating technical systems from network environments”, allowing multiple network modalities to coexist on a unified infrastructure platform. However, existing studies mostly focus on the construction of multimodal network environments, compilation optimization, and network element design, while lacking systematic research on load balancing in the distributed control plane. Some approaches inspired by the switch migration and dynamic reallocation mechanisms of software-defined networking (SDN) can alleviate controller overload to some extent, but they require frequent synchronization of state information among controllers. This leads to high migration overhead and response delays, making it difficult to meet the real-time and scalability requirements of multimodal networks.To address these challenges, this paper proposes a method called joint optimization of routing and polymorphic network element controller allocation (JRECA), which optimizes the control plane load distribu-

收稿日期:2025-05-26;录用日期:2025-11-03;责任编辑:覃怀银
*通讯作者:李传煌

tion in multimodal networks by rationally planning data-plane traffic routing. The method explicitly incorporates the differences in control information scales among different modalities into the optimization framework, comprehensively considering constraints such as network element allocation, routing selection, controller processing capacity, and link bandwidth. Considering the heterogeneous nature of multimodal networks, this paper introduces a load-balancing mechanism that accounts for the differences in control information scale among modalities within the controller load constraints. It constructs a unified model that simultaneously achieves control-plane load balancing and data-plane throughput maximization, thereby realizing coordinated optimization between the control plane load and data-plane throughput—addressing the shortcomings of decoupled optimization in prior research. Furthermore, a theoretically grounded two-step algorithm framework is designed: First, a multimodal network element-controller allocation algorithm based on maximum-load priority is developed to determine the matching relationships between network elements and controllers, with a proven approximation ratio that strictly bounds algorithmic performance. Then, under dynamic traffic conditions, an online routing algorithm based on the primal-dual method is designed, with competitive ratio analysis providing a theoretical lower bound on online optimization performance. Simulation experiments on two representative topologies—Fat-Tree and ARPANet—demonstrate that the proposed algorithm achieves significant performance improvements across five network modalities: IPv4, IPv6, industrial control identifiers, named data identifiers, and identity identifiers. Compared with benchmark algorithms, the proposed method reduces controller load by 17.56% ~ 20.97% and increases system throughput by 13.86% ~ 29.82%.

Key words: polymorphic network; distributed; control plane; load balancing; routing; approximation algorithm

Foundation Item(s): National Natural Science Foundation of China (No.U22A2005); Zhejiang Provincial Key Research and Development Program (No.2024SSYS0001)

1 引言

随着数字化时代垂直行业应用的蓬勃发展,工业互联网、车联网、远程医疗、机器人控制、视频点播等新型服务模式不断涌现,对网络的大带宽、高可靠、低时延、巨连接等核心能力提出了更高要求. 尽管身份标识网络[1]、命名中心网络[2]、Mobility First[3]等新型网络模态在部分行业网络中初步应用并取得良好效果,但是未能从根本上解决多样化网络模态在同一网络设施中的共生及演化问题,无法充分实现新型网络模态的优势. 为此,战略支援部队信息工程大学邬江兴院士团队提出多模态网络环境[4~7]的思路,采用“技术体制与网络环境分离”思想,实现多种网络模态在同一基础网络平台上的共生共存,支持各种网络模态在多模态网络环境上动态加载和运行.

在多模态网络环境中,控制平面由控制器组成,负责网络管理和资源分配. 控制平面通过收集网络的拓扑结构、节点状态、链路带宽等信息获取网络的全局信息,在此基础上对多模态网元节点上网络模态的部署以及流量的路由进行决策,确保不同模态之间能够协调合作,以最大化网络性能. 数据平面由多模态网元节点(如白盒多模态网络)组成,负责各种网络模态数据包的解析与转发,根据控制平面的指令配置转发规则.

当网络拓扑结构不断拓展,控制平面所要管理的多模态网元节点数量持续攀升,同时还需处理日益增长的网络流量. 与软件定义网络(Software Defined Network,SDN)[8]中的OpenFlow[9]协议类似,目前多模态网络采用的流量路由方式,是将每个新到达流的首数据包(即第一个数据包)上传至控制器,再由控制器在多模态网络上被动安装转发规则. 然而,若控制平面仅配备单个控制器,受其处理能力的限制,数据包处理耗时将大幅增加,严重时甚至会造成控制平面拥塞. 由此可见,单个控制器极易成为多模态网络运行的瓶颈,进而降低系统吞吐率. 为规避单个控制器出现拥塞或故障的风险,多模态网络的控制平面应当由多个控制器协同构成. 在分布式控制平面的应用过程中,核心难题在于如何解决因流量动态变化引发的控制器负载失衡问题. 实际情况中,当大量流量涌入某多模态网络,负责管理该多模态网络的控制器便可能出现过载,而其他控制器却可能处于资源闲置状态. 因此,解决控制器负载不均衡问题是确保多模态网络分布式控制平面达到理想性能的关键所在.

虽然目前多模态网络领域已有很多相关工作,但其主要关注于多模态网络环境构造[10~13]、程序编译优化[14~16]、网元节点系统设计[17,18]等方面,尚缺乏分布式控制平面负载均衡相关研究. SDN领域中有部分研究通过引入交换机迁移或动态重分配机制缓解控制器过载问题[19,20],即允许交换机灵活变更所连接的控制器,从初始连接的控制器切换至目标控制器. 但该方案存在显著缺陷,目标控制器需从原控制器处获取新接入多模态网络的详尽信息,这一过程大幅增加了多模态网络迁移耗时,大大降低该方法在多模态网络实际应用场景中的可行性.

为解决多模态网络中分布式控制平面负载均衡的问题,本文采用静态的多模态网元-控制器分配方式.

在此模式下,每台多模态网元与特定控制器保持固定连接状态. 然而,静态分配方式仍需解决流量动态变化带来的控制器负载不均和系统吞吐量下降的问题. 尽管有些工作尝试通过控制器虚拟化与集群化部署[21,22]缓解负载不均,但这些方法仍主要局限于控制平面自身,未能考虑数据平面流量分布情况. 还有些研究工作局限于控制平面的负载优化,未能构建与数据平面的协同优化机制[23,24]. 实际上,控制平面中各个控制器的负载分布情况与数据平面中流量的路由紧密相关. 数据平面中数据包所经的路由路径,会直接决定控制平面中不同控制器的工作负载. 因此,本文尝试通过对数据平面的流量路由进行合理规划,来优化多模态网络中控制平面负载分布,对网络中多种网络模态的流量路由与多模态网元-控制器分配进行联合优化(Joint optimization of Routing and polymorphic network Element Controller Allocation,JRECA),实现控制平面负载均衡与数据平面吞吐最大化. 本文将该问题建模为一个整数规划问题,由于其非凸性质,难以直接求解,因此采取"两步走"方式:首先根据历史流量数据确定控制器与多模态网元之间的匹配关系,实现控制平面负载均衡;然后充分考虑数据平面链路资源约束和控制器处理能力约束,设计基于原始-对偶的在线流量路由方法,实现系统吞吐最大化. 实验结果表明,本文所提算法在系统吞吐率和控制平面负载均衡等方面均具有显著优势.

2 相关工作与研究创新

2.1 相关工作

当前学术界在多模态网络领域的研究主要聚焦于数据平面的优化,如多模态网络环境构造[10~13]、程序编译优化[14~16]、网元节点系统设计[17,18],而对控制平面的优化涉及较少. 具体来说,多模态网络环境构造方面,文献[10,11]提出的多模态网络环境框架,强调通过应用网络与基础设施的分离实现灵活构建与动态演进;文献[12]设计的基于软硬件协同的网元方案,提升了异构资源的利用效率;文献[13]提出的异构标识空间管控架构,优化了空间互通与切换问题. 程序编译优化方面,文献[14]提出的基于P4的转发与加密一体化技术,通过硬件加速和卸载加解密操作,显著提升了系统性能;文献[15]提出的多模态网络编程方案,旨在实现增量式和全维度的网络编程,并重点研究了网络模态生成及安全功能验证;文献[16]则提出支持增量式编程的多模态环境,设计了平台无关的编程模型,提升了开发效率,并有效控制了编译时延和资源开销. 网元节点系统设计方面,文献[17]提出的网络模态增量式部署机制,有效降低了部署复杂度,为多模态网络提供了灵活的部署方式;文献[18]研究的多智能体协同控制框架,通过通信拓扑重构提升了系统协同能力. 但是它们没有考虑到数据平面流量的分布状态对控制平面负载带来的影响,导致多模态网络控制平面存在负载不均的问题.

在SDN领域,部分研究通过引入交换机迁移或动态重分配机制缓解控制器过载问题,即通过将部分交换机切换至其他控制器管理,以实现负载均衡. 例如,文献[19]提出了一种基于多阈值机制的迁移方法,在检测到控制器过载时将部分交换机迁移至其他控制器,以减轻单点压力;文献[20]则设计了基于负载预测的多控制器集群调度算法,通过实时监控触发迁移操作. 但这类方法通常依赖目标控制器获取完整的网络状态,从而引入额外通信与计算开销,并因状态同步增加迁移时延而削弱其实时性与可扩展性. 同时,多模态网络中由于控制信息规模与交互复杂度差异显著,这些问题将被进一步放大. 另有部分研究则聚焦于通过控制器虚拟化与集群化部署提升系统弹性与扩展性,如文献[21]研究了Wi-Fi与LTE-U异构环境下的控制器部署,通过增加逻辑控制器实例分摊负载并增强容错;文献[22]探索了跨空间与地面网络的集成架构,进一步解决集群化控制器部署在异构环境下的可扩展性问题. 但这些工作主要局限于控制平面,未能充分考虑数据平面流量分布对控制器负载的直接影响,从而难以实现两者的全局协同优化. 还有些研究工作局限于控制平面的负载优化,未能构建与数据平面的协同优化机制. 如文献[23]将控制器部署问题抽象为延迟-成本双目标最小化模型,优化焦点完全集中于控制平面的资源配置;文献[24]则提出了一种基于深度强化学习的多路径服务质量拥塞感知路由方法,但该方法的优化目标主要停留在控制平面决策层面,缺乏对数据平面链路带宽约束和流量转发特性的深度耦合. 尽管这类方法在一定程度上缓解了控制器负载不均,但缺乏对数据平面带宽约束与流量转发特性的深度耦合,因此难以满足多模态网络环境下的全局最优配置与性能最大化需求.

2.2 研究创新

本文面向多模态网络提出对分布式控制平面负载均衡与数据平面吞吐量最大化的联合优化方案. 相较既有工作的局限,本文构建了以网元-控制器分配与流量路由为核心的统一建模,并通过"两步走"算法在工程可行性的同时提供近似比与竞争比等严格理论保证.

2.2.1 面向多模态网络场景的控制平面负载均衡方法

当前多模态网络领域针对控制平面负载均衡的研究尚属空白,相关探索主要集中于软件定义网络领

域,而其中现有研究[19,21,23,25,26]大多建立在单一网络模态流量的假设之上. 例如,基于动态迁移的负载均衡策略通常只针对单模态流量的控制器资源分配进行优化,能够通过简单状态同步在单模态环境下有效缓解过载,但在多模态场景中却难以适配. 原因在于,不同模态在控制信息规模(如工控标识流表开销是IPv4的3~5倍)、标识解析规则、处理逻辑及交互数据量等方面差异显著,直接套用迁移类方法会大幅增加状态同步的通信开销与时延,严重削弱方法的实时性与可扩展性. 因此,如何突破传统单模态假设,构建能够适配多模态异构特性的研究框架,成为控制平面负载均衡的核心难点. 针对这一问题,本文提出专门面向多模态网络的控制平面负载均衡方法,将不同模态的控制信息规模差异显式纳入控制器负载约束.

2.2.2　对控制平面负载均衡和数据平面吞吐最大化进行联合优化

现有研究在优化网络性能时往往聚焦于单一目标,例如最小化控制器负载、降低迁移代价或减少路由开销[19,23,24]. 这类单目标导向的方法虽然能够在局部性能指标上取得一定改进,但难以在系统层面兼顾整体效益与均衡性,因而在多模态复杂场景下存在明显局限. 多模态网络中的数据平面路由路径直接影响控制平面的流表下发规模与处理压力,而控制器的负载状态又会反过来限制路由选择空间,二者之间存在紧密耦合关系. 然而,传统研究大多孤立优化某一平面(如仅优化控制器分配或仅优化路由),缺乏有效的协同机制. 针对这一难点,本文提出将数据平面吞吐量最大化与控制平面负载均衡纳入统一优化框架,构建JRECA联合优化模型,同时优化控制平面的负载与数据平面的流量路由.

2.2.3　具有严格理论保障的算法设计

本文将JRECA建模为整数规划问题. 由于该问题包含二元变量及乘积项而呈现非凸特性,且多模态网元-控制器分配问题本身可归约为并行机调度问题,因此JRECA属于NP-hard(Non-deterministic Polynomial-time hard)问题. 以往研究[25,26]多采用遗传算法、蚁群算法等启发式方法求解此类问题,虽能获得可行解,但缺乏严格的理论性能保证. 针对这一难点,本文提出具有严格理论保障的“两步走”算法框架:首先,设计基于最大负载优先的多模态网元-控制器分配算法,确定多模态网元与控制器之间的匹配关系,并通过近似比证明严格界定了算法性能边界;然后,在动态流量环境下,设计基于原始-对偶方法的在线路由算法,并通过竞争比分析给出在线优化的理论性能下界. 与传统启发式方法不同,该框架在理论上兼具近似比与竞争比的严格保证.

3　系统模型及问题定义

3.1　系统模型

现有工作[19,21,23,25,26]大多建立在所有流量均属于一种网络模态的假设之上. 然而,在多模态网络背景下,不同模态的控制信息规模(流表下发开销)和数据量等方面存在显著差异,导致现有方法在多模态网络环境中难以取得理想效果. 本文JRECA问题建模部分则专门面向多模态网络设计,如图1所示.

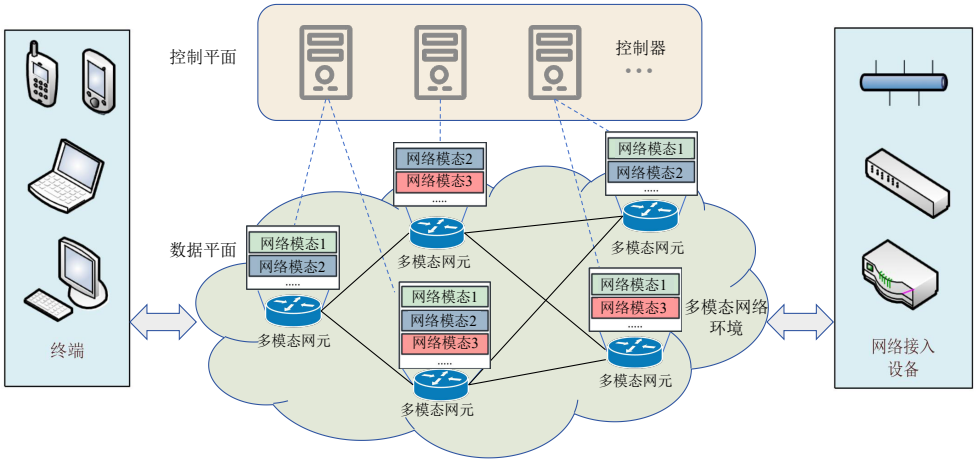


图1　多模态网络环境

多模态网络由控制平面和数据平面组成. 控制平面包含逻辑集中的控制器,负责资源管理、流量调度等任务. 数据平面由多模态网元以及链路组成,负责根据控制平面的指令对多种网络模态的数据包进行转发. 本文使用c表示控制器,使用C表示控制平面中的控制器集合. 将数据平面建模成一个有向无环图$G=$

$(V\cup E)$,其中e表示链路,E表示链路集合,v表示多模态网元,V表示多模态网元集合.链路负责将终端和多模态网元连接起来,将链路$e\in E$的带宽容量记为C_e,控制器$c\in C$的处理能力使用C_c表示.设多模态网络中的网络模态集合为M,多模态网络中存在多种网络模态的流量,将网络模态为$m\in M$的流量集合表示为Γ_m,网络中所有网络模态的流量集合表示为$\Gamma=\bigcup_{m\in M}\Gamma_m$.每一条流量$\gamma\in\Gamma$可由三元组$(u_s,u_d,m)$表示,其中$u_s\in U$表示流$\gamma$的源终端,$u_d\in U$表示流$\gamma$的目的终端,$m$表示流$\gamma$所属的网络模态.设流$\gamma$的流量大小为$f(\gamma)$.考虑到不同模态的控制信息规模(流表下发开销)和数据量等方面存在显著差异,本文使用二元常量φ_m^γ表示网络模态为m的流量γ下发控制流表的信息大小.表1总结了本文使用的参数符号.

表1 参数符号表

参数	含义
U	终端集合
E	链路集合
Γ	流量集合
M	网络模态集合
V	多模态网元集合
C	控制器集合
C_c	控制器c的处理能力
K_c	控制器c的容量
C_e	链路e的带宽容量
P_γ	流γ的候选路径集合
φ_m^γ	表示流量γ模态m流表下发信息大小
$f(\gamma)$	流γ的流量大小
x_v^c	二元变量,表示控制器c是否连接多模态网元v
y_γ^p	二元变量,表示流γ是否选择$p\in P_\gamma$作为路由路径

3.2 问题定义

本文尝试从流量路由的角度来优化多模态网络中控制平面负载分布,对网络中多种网络模态的流量路由与多模态网元-控制器分配进行联合优化JRECA.具体来说,在给定控制器数量、多模态拓扑和终端分布以及流量集合的前提下,JRECA需要解决以下两个子问题:一是每个控制器负责管理哪些多模态网元;二是每条流走哪条路由路径来将数据包转发至相应的目的终端.本文使用二元变量x_v^c表示控制器c是否负责多模态网元v的流量处理.若多模态网元v被控制器c管理,则$x_v^c=1$;否则,$x_v^c=0$.每条流γ有一个候选路径集合P_γ,可通过K-最短路径算法[27]得到该集合.使用$f(\gamma)$表示流γ的流量大小,使用二元变量y_γ^p表示流γ是否选择$p\in P_\gamma$作为路由路径.若流γ的路由路径为p,则$y_\gamma^p=1$;否则,$y_\gamma^p=0$.总而言之.JRECA需要在给定拓扑和流量集合下解决上述两个子问题,确定变量x_v^c与y_γ^p的值,以完成多模态网元的分配和流量的路由.

本文在对流量路由与多模态网元-控制器分配联合优化时,充分考虑以下约束:

(1)分配约束.对于任意一个多模态网元而言,需为其分配一个控制器进行管理,如式(1)所示:

$$\sum_{c\in C}x_v^c=1,\forall v\in V \tag{1}$$

(2)选路约束.对于任意一条流量γ而言,需为其选择一条路由路径进行转发,如式(2)所示:

$$\sum_{\gamma\in P_\gamma}y_\gamma^p\leqslant 1,\forall\gamma\in\Gamma \tag{2}$$

(3)控制器资源约束.在控制器对多模态网络进行流表下发时,不应超出控制器拥有的资源容量,如式(3)所示:

$$\sum_{\gamma\in\Gamma}\sum_{v\in p:p\in P_\gamma}y_\gamma^p\cdot x_v^c\cdot\varphi_m^\gamma\leqslant K_c,\forall c\in C \tag{3}$$

(4)链路资源约束.每条链路上经过的总流量大小不应超出该链路的带宽容量,如式(4)所示:

$$\sum_{\gamma\in\Gamma}\sum_{e\in p:p\in P_\gamma}y_\gamma^p\cdot f(\gamma)\leqslant C_e,\forall e\in E \tag{4}$$

在优化目标上,已有研究多聚焦于单一指标,如降低控制器负载[19]、最小化延迟与控制开销[23],或减少转发代价[24],虽能在局部性能上取得改进,但缺乏系统整体效益的刻画.但在多模态网络中,实现控制平面的负载均衡是保障多模态网络稳定高效运行的关键措施.同时,系统吞吐量是衡量网络性能的核心指标,直接影响网络资源利用效率与服务响应速度.因此,本文将实现控制平面的负载均衡和最大化系统吞吐量作为优化目标,使用权重系数α、β来调整这两个部分在联合目标中的相对影响力,如式(5)所示:

$$\max\alpha\cdot\sum_{\gamma\in\Gamma}\sum_{p\in P_\gamma}y_\gamma^p\cdot f(\gamma)-\beta\cdot\sum_{\gamma\in\Gamma}\sum_{v\in p:p\in P_\gamma}y_\gamma^p\cdot x_v^c\cdot\varphi_m^\gamma \tag{5}$$

综上所述,JRECA问题可被形式化如式(6)所示:

$$\begin{aligned}&\max\alpha\cdot\sum_{\gamma\in\Gamma}\sum_{p\in P_\gamma}y_\gamma^p\cdot f(\gamma)-\beta\cdot\sum_{\gamma\in\Gamma}\sum_{v\in p:p\in P_\gamma}y_\gamma^p\cdot x_v^c\cdot\varphi_m^\gamma\\&\text{s.t.}\begin{cases}\text{约束:}(1)\sim(4)\\x_v^c\in\{0,1\},\forall v\in V\\y_\gamma^p\in\{0,1\},\forall\gamma\in\Gamma,p\in P_\gamma\end{cases}\end{aligned} \tag{6}$$

4 路由与多模态网元-控制器分配联合优化算法设计

本文选用基于最大负载优先的多模态网元-控制器分配算法和基于原始-对偶方法的实时路由算法,其选择依据主要基于问题特性与算法的理论性能保证.由于公式中的约束条件出现了二元变量x_v^c和y_γ^p的乘积,以及变量x_v^c和y_γ^p存在二元特性,因此多模态网元-控制

器匹配问题属于NP-hard问题,难以直接求解. 传统的启发式或元启发式方法(如遗传算法、粒子群优化等)虽然能够在有限时间内给出可行解,但普遍存在计算复杂度较高、收敛性依赖参数调优、缺乏严格理论性能保证等问题,在大规模网络或高动态性的在线应用场景下难以满足实时性与可控性需求. 基于此,本文设计的路由与多模态网元-控制器分配联合优化算法(Joint Optimization Algorithm for Routing and polymorphic network element-controller Allocation, JOARA)由以下两个步骤组成:第一步,根据历史流量确定多模态网元与控制器之间的匹配关系,使各个控制器的负载均衡;第二步,考虑链路资源以及控制平面负载的情况,对流量进行实时路由,实现系统吞吐量最大化. 相比于传统的启发式或元启发式方法,最大负载优先策略的分配算法在理论上具备近似比保证,即在最坏情况下其解与最优解之间的性能差距可被严格界定. 这一性质使得算法即便在极端条件下也能提供可预测的性能边界,从而为多模态网络控制平面的负载分配提供了更加可靠和可验证的理论保障.

4.1　多模态网元-控制器分配算法

本节介绍求解JRECA问题的第一步算法,即多模态网元-控制器分配算法. 该算法需确定每个控制器需要管理哪些多模态网元,即确定多模态网元与控制器的匹配关系,以使得各个控制器的负载相近. 定义控制器负载因子λ,表示控制器资源消耗比例. 将多模态网元-控制器匹配问题形式化如式(7)所示:

$$\min \lambda \quad \text{s.t.} \begin{cases} \sum\limits_{c \in C} x_v^c = 1, \forall v \in V \\ \sum\limits_{\gamma \in \Gamma} \sum\limits_{v \in P_\gamma} x_v^c \cdot \varphi_m^\gamma \leqslant C_c \cdot \lambda, \forall c \in C \\ x_v^c \in \{0,1\}, \forall v \in V, c \in C \end{cases} \tag{7}$$

其中,第一个约束表明每个多模态网元必须为其分配一个控制器进行管理,第二个约束为控制器负载均衡约束,优化目标为最小化控制器的最大负载,即$\min \lambda$.

定理1　多模态网元-控制器匹配问题是NP-hard问题.

证明　为了证明多模态网元-控制器匹配问题是NP-hard问题,本文先给出如下定义.

定义:并行机调度(Identical Parallel Machines Scheduling, IPMS)问题[28]. 给定m个并行机器和n个独立作业,每个作业都将被分配给具有相同处理速度的机器之一. 因此,每个作业在所有机器上花费相同的处理时间. 该问题的目标是在合适的机器上安排工作,以尽量减少总完成时间.

由于IPMS是NP-hard问题,可通过证明IPMS问题是多模态网元-控制器匹配问题的一个特例来证明多模态网元-控制器匹配问题为NP-hard问题. 考虑所有控制器的处理能力都相同,变量x_v^c表示多模态网元v是否由控制器c管理,将其类比为任务是否被分配到某个机器中. 式(7)中的第一个约束表示多模态网元仅被分配给一个控制器,可类比为每个任务只能分配到一个机器;第二个约束为控制器处理能力约束,可类比为处理器的处理能力约束. 式(7)的优化目标$\min \lambda$对应IPMS最小化整体完成时间. 因此IPMS问题是多模态网元-控制器匹配问题的一个特例. 由于IPMS问题是NP-hard问题,因此多模态网元-控制器匹配问题也是NP-hard问题. 证毕.

为解决多模态网元-控制器匹配问题,本文参考求解IPMS问题的最长处理时间优先(Longest Processing Time, LPT)算法[29],提出了一种基于多模态网元最大负载优先的算法来实现控制器的负载均衡. 该算法的核心思想是在考虑控制器处理能力限制的前提下,根据多模态网元消耗的控制资源量对其进行排序,优先分配资源需求大的多模态网元到当前负载较低的控制器,以达到负载均衡的目标. 具体来说,该算法首先计算每个多模态网元v的控制资源需求量R_v,并根据R_v对所有多模态网元进行排序. 然后,将每个控制器的负载L_c初始化为0. 最后,逐一为排序后的多模态网元分配负载最小的控制器,在分配的同时确保每个控制器的负载不超过其处理能力. 算法的详细过程如算法1所示.

定理2　算法1取得$4/3-1/\left(3|C|\right)$的近似比(其中$|C|$为控制器数量),即控制器的最大负载最多是最优解的$4/3-1/\left(3|C|\right)$倍.

由于定理2的证明过程与求解IPMS问题的LPT算法的近似比证明过程相似,因此此处不再赘述.

4.2　实时流量路由算法

本节介绍求解JRECA问题的第二步算法,即实时流量路由算法. 在第一步中,我们根据历史流量信息确定了多模态网元与控制器之间的匹配关系,然而由于流量的动态性,控制器仍然会出现过载的情况,因此在对流量进行路由时仍需考虑控制器的负载情况. 将多种网络模态流量的实时路由问题形式化如式(8)所示:

$$\max \sum_{\gamma \in \Gamma} \sum_{p \in P_\gamma} y_\gamma^p \cdot f(\gamma)$$

$$\text{s.t.} \begin{cases} \sum\limits_{\gamma \in P_\gamma} y_\gamma^p \leqslant 1, \forall \gamma \in \Gamma \\ \sum\limits_{\gamma \in \Gamma} \sum\limits_{v \in p: p \in P_\gamma} y_\gamma^p \cdot \varphi_m^\gamma \cdot I(v,p) \leqslant C_c, \forall c \in C \\ \sum\limits_{\gamma \in \Gamma} \sum\limits_{e \in p: p \in P_\gamma} y_\gamma^p \cdot f(\gamma) \leqslant C_e, \forall e \in p \\ y_\gamma^p \in \{0,1\}, \forall \gamma \in \Gamma, p \in P_\gamma \end{cases} \tag{8}$$

其中,第一个约束为选路约束,即每条流量至多选取一条路径进行路由;第二个约束为控制器处理能力约束;第三个约束为数据平面的链路约束. 优化目标为最大化系统吞吐量.

算法1 基于最大负载优先的多模态网元-控制器分配算法

输入:多模态网元集合V、控制器集合C和控制信息大小φ_m^γ、每条流γ的候选路径集合P_γ

输出:控制器分配变量x_v^c的整数解$\hat{x}_v^c$

步骤一:计算多模态网元控制资源需求

FOR $v \in V$:

设多模态网络总控制资源需求为S_v

获得多模态网元v的控制资源需求$R_v=\sum_{\gamma\in\Gamma:v\in P_\gamma}\varphi_m^\gamma$

END FOR

将多模态网元$v \in V$按照R_v降序排序

步骤二:初始化控制器负载

FOR $c \in C$:

设控制器负载为L_c

初始化$L_c=0$

END FOR

步骤三:多模态网元分配

FOR $v \in V$:

选择当前负载最小的控制器$c_{\min}=\min(L_c)$

IF $L_{c_{\min}}+R_v \leqslant C_c$:

将$\hat{x}_v^{c_{\min}}$置1

$L'_{c_{\min}} \leftarrow L_{c_{\min}}+R_v$

ELSE:

FOR $c \in C$:

IF $L_c+R_v \leqslant C_c$:

将$\hat{x}_v^c$置为1

$L'_c \leftarrow L_c+R_v$

BREAK;

END IF

END FOR

END IF

END FOR

计算最大控制负载$\lambda=\max(L_c),\forall c\in C$

为解决式(8)中的问题,本文设计了一种基于原始对偶方法的实时流量路由算法. 首先为线性松弛问题构造对偶问题,令α_γ、β_c和δ_e分别为第一组、第二组和第三组不等式的对偶变量,且所有对偶变量都非负. 该对偶问题可表述为式(9):

$$\min \sum_{\gamma\in\Gamma}\alpha_\gamma+\sum_{v\in V}\beta_c C_c+\sum_{e\in p}\delta_e C_e$$
$$\text{s.t.}\begin{cases}\alpha_\gamma \geqslant f(\gamma)-\sum_{v\in V}\beta_c\cdot\varphi_m^v\cdot I(v,p)\\ \quad -\sum_{e\in p}\delta_e\cdot f(\gamma),\forall\gamma\in\Gamma,v\in p\\ \alpha_\gamma\geqslant 0,\beta_c\geqslant 0,\delta_e\geqslant 0\end{cases}\tag{9}$$

本文根据对偶问题来构建式(8)定义的原始问题的解. 算法的第一步为初始化所有对偶变量以及常数U^*、N和ε. 其中常数U^*表示所有路由方案中每种资源的最大使用量,计算公式为式(10):

$$U^*=\max_{\gamma,v,e}\left\{\max_{\gamma,v}\frac{\varphi_m^\gamma\cdot I(v,p)}{f(\gamma)},\max_{\gamma,e}f(\gamma)\right\}\tag{10}$$

常数N表示式(8)中第二组和第三组不等式数量的多模态网元数量,即$N=2\cdot|V|$,常数$\varepsilon\in(0,1)$表示资源消耗的奖惩程度.

算法的第二步是为每条流量从候选路径集合P_γ中选取一条路径P进行路由,并根据每条候选路径P的收益I_p选择路由路径. 计算路径收益时,并未采用人工预设固定权重的方式,而是通过原始-对偶优化框架,综合考虑流量大小、控制器资源消耗和链路资源消耗等关键因素,动态生成自适应性路径收益. 路径收益I_p的计算公式为式(11):

$$I_p=f(\gamma)-\sum_{v\in V}\beta_c\cdot\varphi_m^\gamma\cdot I(v,p)-\sum_{e\in p}\delta_e\cdot f(\gamma)\tag{11}$$

其中,$f(\gamma)$表示流γ的流量大小,是收益计算的基础基准. $\sum_{v\in V}\beta_c\cdot\varphi_m^\gamma\cdot I(v,p)$反映流$\gamma$经过的多模态网元所关联控制器的资源占用代价,其中$\beta_c$为控制$c$的对偶变量,作为控制器资源约束的动态权重. φ_m^γ为流γ所属模态m的控制信息大小其为常量,由模态特性决定,如IPv4/IPv6模态的流表下发开销差异. $\sum_{e\in p}\delta_e\cdot f(\gamma)$反映路径$P$上链路资源的占用代价,其中$\delta_e$为链路$e$的对偶变量,作为链路资源约束的动态权重. 对偶变量β_c(控制器权重)与δ_e(链路权重)会通过式(12)(13)自适应迭代更新. 链路越拥堵、控制器负载越高,其对应的对偶变量值越大,路径收益计算时的惩罚权重也越大,从而引导流量避开资源紧张的链路与控制器,实现负载均衡与吞吐量优化的自适应协同.

使用P^*表示所有路径中具有收益最大的路径,$I_{\max}$表示P^*的收益值. 如果$I_{\max}\leqslant 0$,这将违反式(9)中对偶可行性,此时流量γ不会被处理. 如果$I_{\max}\geqslant 0$,算法根据路径P^*对流量γ进行路由,并根据式(11)更新α_γ,对于控制器c以及路径e,其对偶变量的更新规则如式(12)(13)所示:

$$\beta'_c=\beta_c\left(1+\frac{\varphi_m^\gamma\cdot I(v,p)}{C_c}\right)+\varepsilon\cdot\frac{\varphi_m^\gamma\cdot I(v,p)}{N\cdot C_c\cdot U^*}\tag{12}$$

$$\delta'_e=\delta_e\left(1+\frac{f(\gamma)}{C_e}\right)+\frac{\varepsilon}{N\cdot C_e\cdot U^*}\tag{13}$$

其中,β_c'和δ_e'分别表示流量γ占用资源后β_c和δ_e的值. 实时流量路由算法的具体过程如算法2所示.

定理3 对于所有对偶变量,算法2中的更新规则是对偶可行的.

算法2　基于原始对偶方法的实时流量路由算法

输入:多模态网元集合V、控制器集合C和控制信息大小φ_m^γ、每条流γ的候选路径集合P_γ

输出:流量路径选择集合P_I

步骤一:算法初始化

根据式(10)初始化常量U^*

初始化常量$N=2\cdot|V|,\varepsilon\in(0,1)$

初始化对偶变量:$\alpha_\gamma=0,\beta_c=0,\delta_e=0,\forall\gamma\in\Gamma,c\in p$

步骤二:确定流量路由方案

FOR $\gamma\in\Gamma$:

根据式(11)计算流量$\gamma\in\Gamma$在路径$p_\gamma\in P$下的收益;

使用P^*表示所有路径中具有收益最大的路径,用$I_{\max}$来表示P^*的收益值,此时$I_{\max}=\max\limits_{\gamma\in\Gamma;p_\gamma\in P} I_{P_\gamma}$

IF $I_{\max}\leqslant 0$:

丢弃流量γ处理,更新$\alpha_\gamma\leftarrow 0$

ELSE:

根据$I_{\max}$方案分配路径$P_I\leftarrow\arg\max\limits_{\gamma\in\Gamma;p_\gamma\in P} I_{p_\gamma}$,对流量$\gamma$采用路径$P^*$进行路由转发

更新α_γ为$I_{\max}$

根据式(12)、(13)更新β_c、δ_e

证明　在算法执行过程中,所有对偶变量α_γ、β_c和δ_e初始化为0,并且根据算法的更新规则,每次迭代时对偶变量的值只会增加,确保它们始终非负. 更新过程是通过加法进行的,增量为非负数,因此对偶变量始终保持非负性,并且每次更新时对偶变量的值都从0开始逐步增大. 此外,更新后的对偶变量不会导致原始问题的约束失效,反而使得目标函数的右边变得更小,从而保证对偶约束始终被满足. 因此,所有对偶变量在算法过程中始终保持非负,并且更新过程保证了对偶约束的可行性. 证毕.

在线算法的竞争比(Competitive Ratio)[30]用于衡量其在未知未来输入下相较最优离线算法的最坏性能比值,并可扩展至资源约束放宽情形. 作为在线近似算法设计与评估的核心指标,竞争比已广泛应用于网络路由、资源调度与任务分配等领域,如云计算资源分配[22]与服务功能链调度[31]. 鉴于多模态网络中"流量动态到达且未来模态与规模不可预知"的特性,本文采用竞争比对所提算法进行性能分析. 下面给出竞争比定义.

定义:若一个在线算法实现的结果至少为$\zeta\cdot$OPT,其中OPT表示最优解的结果,且约束被违反η倍,则称该在线算法具有$[\zeta,\eta]$的竞争比.

定理4　算法2的实现结果至少为$(1-\varepsilon)\cdot$OPT,其中OPT表示最优解的结果.

证明　在算法2中,当流量γ被处理时,算法将使优化目标的值增加,根据算法中的对偶变量更新规则,由式(11)~(13)得到优化目标值增加,如式(14)所示:

$$\begin{aligned}\Delta&=f(\gamma)-\sum_{v\in V}\beta_c\cdot\varphi_m^v\cdot I(v,p)-\sum_{e\in p}\delta_e f(\gamma)\\&+\sum_{v\in V}\left[\beta_c\left(1+\frac{\varphi_m^\gamma\cdot I(v,p)}{C_c}\right)+\varepsilon\cdot\frac{\varphi_m^\gamma\cdot I(v,p)}{N\cdot C_c\cdot U^*}\right]\\&+\sum_{e\in p}\left[\delta_e\left(1+\frac{f(\gamma)}{C_e}\right)+\frac{\varepsilon}{N\cdot C_e\cdot U^*}\right]\\&\leqslant f(\gamma)+\frac{2\cdot\varepsilon\cdot f(\gamma)}{N\cdot U^*}=(1+\varepsilon)\cdot f(\gamma)\end{aligned}\tag{14}$$

这意味式(9)对应的对偶问题的目标值最多增加$(1+\varepsilon)\cdot f(\gamma)$. 因此,式(8)对应原问题的优化目标整体值至少是最优解的$1/(1+\varepsilon)\geqslant 1-\varepsilon$倍. 最终,算法2能实现$(1-\varepsilon)\cdot$OPT倍的吞吐量. 证毕.

定理5　对于每一条流量γ和多模态网元v,都满足式(15):

$$\begin{aligned}\beta(v,\gamma,c)&\geqslant\varepsilon\cdot\frac{\exp\left[G(v,\gamma)/K_c\right]-1}{N\cdot U^*}\\\delta(v,\gamma,e)&\geqslant\varepsilon\cdot\frac{\exp\left[H(v,\gamma)/C_e\right]-1}{N\cdot U^*}\end{aligned}\tag{15}$$

证明　本文通过数学归纳法,来证明式(15)的第一个不等式. 对于每一条流量γ和路径中的多模态网元节点v,$G(v_i,\gamma)$表示在第i次迭代时,流量γ流经某条路径到达多模态网元v后链路的负载. 假设初始时刻网络负载为0,即$G(v_0,\gamma)=0=\beta(v_0,\gamma,c)$,这表示在开始时网络的负载为0,所以不等式显然成立. 我们假设当$i=i-1$时,归纳假设成立,如式(16)所示:

$$\beta(v_{i-1},\gamma,c)\geqslant\varepsilon\cdot\frac{\exp\left[G(v_{i-1},\gamma)/K_c\right]-1}{N\cdot U^*}\tag{16}$$

接下来我们考虑第i步的情况. 根据算法的更新规则,每当一个请求γ到达时,它将对负载$G(v_{i-1},\gamma)$产生影响. 如果请求没有被拒绝,则负载会增加,更新后的负载$G(v_i,\gamma)$如式(17)所示:

$$G(v_i,\gamma)=G(v_{i-1},\gamma)+\varphi_m^\gamma\cdot I(v,p)\tag{17}$$

其中,φ_m^γ为表示流量γ模态m流表下发信息大小,$I(v,p)$为多模态网元v选择路径p的收益.

因此,在第i步$\beta(v_i,\gamma,c)$也会更新,根据算法的更新规则,新的$\beta(v_i,\gamma,c)$可以表示为式(18):

$$\beta(v_i,\gamma,c)=\beta(v_{i-1},\gamma)\cdot\left[1+\frac{\varphi_m^\gamma\cdot I(v,p)}{K_c}\right]+\varepsilon\cdot\frac{\varphi_m^\gamma\cdot I(v,p)}{N\cdot K_c\cdot U^*}\tag{18}$$

其中,K_c表示控制器c的容量

我们将归纳假设应用到上面的更新公式中,假设在第$i-1$步归纳假设成立,如式(19)所示:

$$\beta(v_{i-1},\gamma,c)\geqslant\varepsilon\cdot\frac{\exp\left[G(v_{i-1},\gamma)/K_c\right]-1}{N\cdot U^*}\tag{19}$$

通过上述的更新公式，得到式(20)：

$$\begin{aligned}\beta(v_{i-1},\gamma,c)\geqslant&\varepsilon\cdot\frac{\exp\left[G(v_{i-1},\gamma)/K_c\right]-1}{N\cdot U^*}\cdot\left[1+\frac{\varphi_m^{\gamma}\cdot I(v,p)}{K_c}\right]\\&+\varepsilon\cdot\frac{\varphi_m^{\gamma}\cdot I(v,p)}{N\cdot K_c\cdot U^*}\\=&\varepsilon\cdot\frac{1}{N\cdot U^*}\cdot\left\{\exp\left[G\left(v_{i-1},\gamma\right)/K_c\right]\left[1+\frac{\varphi_m^{\gamma}\cdot I(v,p)}{K_c}\right]-1\right\}\\\approx&\varepsilon\cdot\frac{1}{N\cdot U^*}\cdot\left\{\exp\left[G\left(v_{i-1},\gamma\right)/K_c\right]\exp\left[\frac{\varphi_m^{\gamma}\cdot I(v,p)}{K_c}\right]-1\right\}\\=&\varepsilon\cdot\frac{\exp\left[G(v_{i-1},\gamma)/K_c\right]-1}{N\cdot U^*}\end{aligned}\tag{20}$$

在上述推导中，对于小的正值x，应用了一阶近似$\exp(x)\approx 1+x$. 式(15)中的第二个等式证明过程同第一个等式，在此不再赘述. 通过递推和归纳法，证明了在每一轮更新之后，不等式依然成立. 证毕.

定理 6 对于任意给定的$\varepsilon\in(0,1)$，算法 2 具有$\left[1-\varepsilon,O\left(\log 2|V|+\log\left(1/\varepsilon\right)\right)\right]$的竞争比.

证明 当流量γ被处理时，式(8)的目标值会随$f\left(\gamma\right)$增加. 根据定理4，对偶问题的目标值增加量小于$(1-\varepsilon)\cdot f\left(\gamma\right)$. 因此，式(8)的整体目标值至少是式(9)的目标值的$1-\varepsilon$倍. 由此可知，基于对偶问题的算法2的资源容量最少是$(1-\varepsilon)\cdot\mathrm{OPT}$.

根据式(12)，在算法2的任意一次迭代中都有$\beta_c\leqslant 1+U^*$. 结合上文分析，能够得到式(21)：

$$\begin{aligned}\frac{G\left(v,\gamma\right)}{C_c}&\leqslant\log\left[\frac{\beta\left(v,\gamma\right)\cdot N\cdot U^*}{\varepsilon}+1\right]\\&\leqslant\log\left[\frac{\left(1+U^*\right)\cdot N\cdot U^*}{\varepsilon}+1\right]\\&\leqslant O\left(\log 2|V|+\log\left(1/\varepsilon\right)\right)\end{aligned}\tag{21}$$

这一结果表明，在算法2结束时，链路资源容量上限为$O\left(\log 2|V|+\log\left(1/\varepsilon\right)\right)$. 因此，本文提出的基于对偶问题的算法2能够达到$\left[1-\varepsilon,O\left(\log 2|V|+\log\left(1/\varepsilon\right)\right)\right]$的竞争比. 证毕.

5 性能评估

在实验部分，本文将JOARA算法与其他算法进行对比，实验结果表明：

(1)JOARA算法在不同流量情况下的吞吐量皆优于对比算法. 例如，在Fat-Tree拓扑中，较基于扰动领导者的成本最小化控制器部署算法(Cost-Minimizing Controller Deployment algorithm based on perturbed leaders, CMCD)和基于贪心策略的控制器部署与交换机分配算法(greedy strategy-based Controller Placement and Switch Assignment algorithm, CPSA)，JOARA算法可将吞吐量分别提升13.86%和29.82%.

(2)JOARA算法的控制器负载因子显著低于对比算法. 以控制器数量为4的场景为例，在Fat-Tree拓扑中，JOARA算法的控制器负载因子相比CMCD算法和CPSA算法分别降低20.79%和32.51%.

5.1 实验环境设置

在实验设置方面，本文算法程序使用Python 3.8进行编写，实验运行于一台配置了Intel Core i9-14900K处理器、128 GB内存、两块NVIDIA GeForce RTX 4090显卡的服务器上，该服务器使用的操作系统为Ubuntu20.04. 网络拓扑主要采用Fat-Tree[32]和ARPA-Net[33]这两个拓扑，并生成以下5种网络模态的流量：IPv4模态、IPv6模态、工控标识模态、命名数据标识模态、身份标识模态. 流量生成方法为2-8原则，即20%的流为大象流，80%的流为老鼠流[34]. 在网络环境中，链路带宽统一为10 Gbps[35]. 在流量控制开销建模时，每条流的模态控制信息随机取自64 ~ 192 Byte区间.

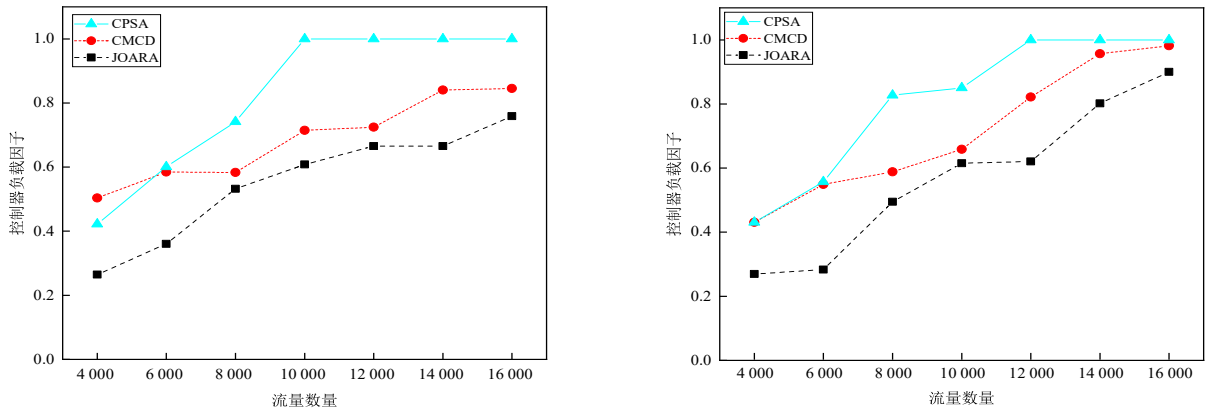
在多模态网络这一研究领域，经过广泛且深入的调研，发现目前尚无专门针对多模态网络控制平面负载均衡相关论文(详见第2节). 鉴于此，为了全面且有效地验证所提算法的有效性与高效性，本文对现有SDN领域中的控制平面负载均衡工作进行修改，从而构建出对比算法. 实验中，本文采用以下两个对比算法：(1)CMCD算法[36]将数据路径选择与控制器部署联合建模为最小化总成本问题，成本包括路由链路开销和控制器到交换机的通信延迟. 为应对动态环境中的链路成本和延迟不确定性，采用基于扰动领导者的在线学习策略(Follow-The-Perturbed-Leader, FTPL)更新策略，通过历史损失和随机扰动优化路径和控制器位置. (2)CPSA算法[37]采用贪心算法联合优化控制器部署与多模态网络分配. 该方法将控制器视为有限容量中心节点，多模态网络为计算任务终端，整体建模为带容量约束的优化问题，使用贪心启发与固定容量模型模拟其分配逻辑，在静态网络结构中复现控制器负载控制能力.

在算法参数设置方面，对于候选路径数，我们在{2,4,8}的范围内进行对比，兼顾路径多样性与运行效率；在JOARA算法的路径筛选阶段，本文设置了剪枝阈值，范围是{0.85,0.90,0.95}，用于控制候选路径数量与搜索深度，从而平衡路径多样性与计算开销；在CPSA算法的迭代优化过程中，本文设置了批量更新步长，范围是{8,16,32}，并在不同取值下进行对比，以考察其在性能与效率之间的权衡. 实验中每组条件均独立重复100次，并统计均值与置信区间，确保结论稳健. 本文对三种算法(JOARA、CMCD、CPSA)均进行了多组对

比实验,并在保证约束条件的前提下,选取了性能最优或近似最优的参数作为最终设定.

5.2 实验结果

本文在两个拓扑中,对流量数量从4 000~16 000条的不同规模进行多次实验,观察流量数量对控制平面负载的影响. 实验结果如图2所示. 图2展示了3种算法在控制器数量为4个时,控制器负载因子随流量变化的情况. 结果表明,随着流量从4 000条增加至16 000条,所有算法的控制器负载因子均呈上升趋势,反映了控制平面负载压力的逐渐增大. 特别是CPSA算法,在Fat-Tree拓扑中于10 000条流量时达满负载,在ARPANet拓扑中于12 000条流量时达到满负载. 而JOARA算法和CMCD算法则维持相对均匀的增长,这部分得益于算法在路径选择上的优化. 对比发现,JOARA算法的控制器负载因子低于CMCD算法,在Fat-Tree拓扑中最高为0.76,在ARPANet拓扑中最高为0.89,并显著低于CPSA算法,与CMCD和CPSA算法相比,JOARA算法的控制器负载因子分别降低约17.56%和20.97%.

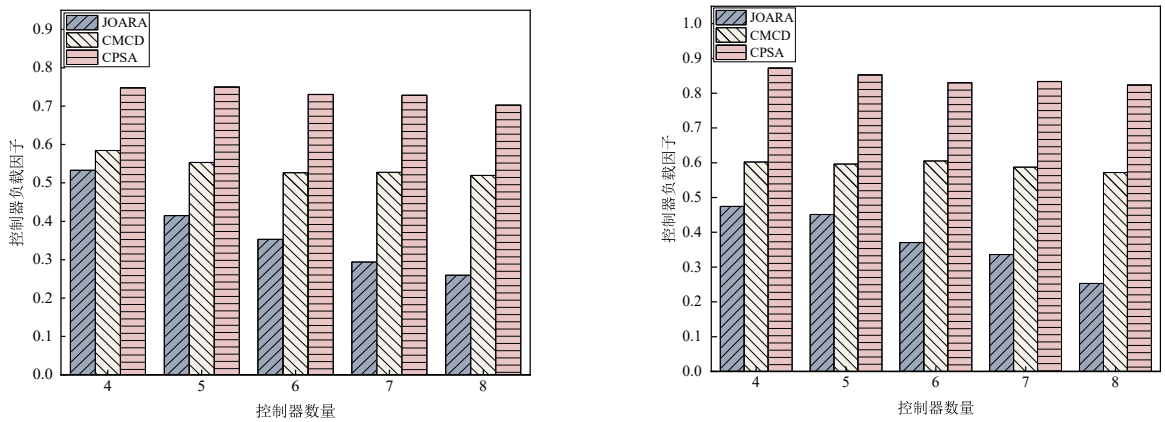


(a) Fat-Tree拓扑　　(b) ARPANet拓扑

图2　控制器负载因子vs流量数量

本文还对比了不同控制器数量下的控制器负载表现. 图3展示了3种控制器部署算法在8 000条流量下,在两个拓扑中控制器负载因子随控制器数量变化的情况. CPSA算法在每次部署时,控制器数量不随总数增加而变化,因此始终保持较高的负载;而CMCD算法在4~8个控制器之间,仍然选择个别数量的控制器进行部署,未能充分利用所有控制器资源,导致负载因子出现轻微下降趋势. 相比之下,JOARA算法的控制器负载因子明显低于对比算法,并且随着控制器数量的增加呈现出显著下降趋势,表明其能够合理分配控制器资源,降低控制器负载,实现了负载均衡的目标.

本文针对流量数量从4 000~16 000条分别在两个网络拓扑中进行多次实验,观察流量数量对网络拓扑中吞吐量的影响,实验结果如图4所示. 在两个拓扑中,随着流量数量的增加,3种算法的吞吐量均呈上升趋势,表明网络成功传输了更多的数据,逐渐接近链路

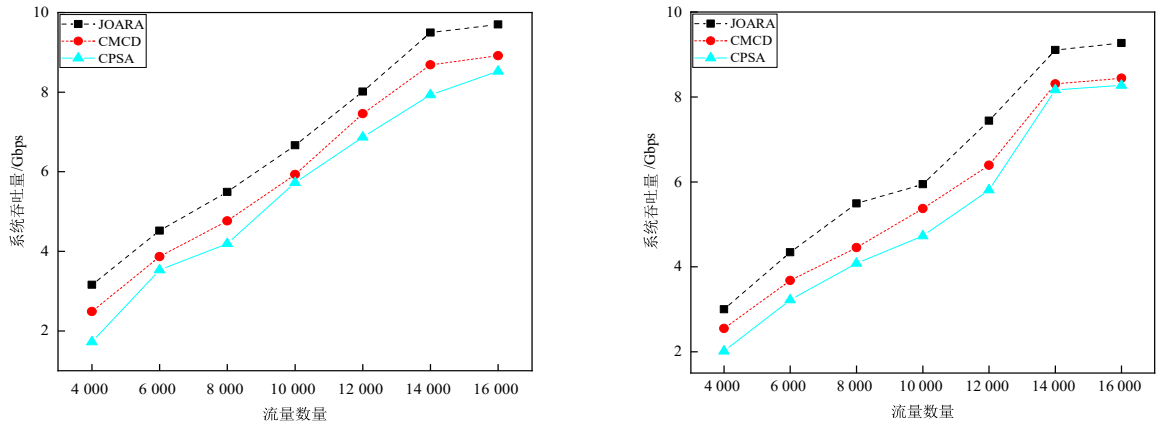


(a) Fat-Tree拓扑　　(b) ARPANet拓扑

图3　控制器负载因子vs控制器数量

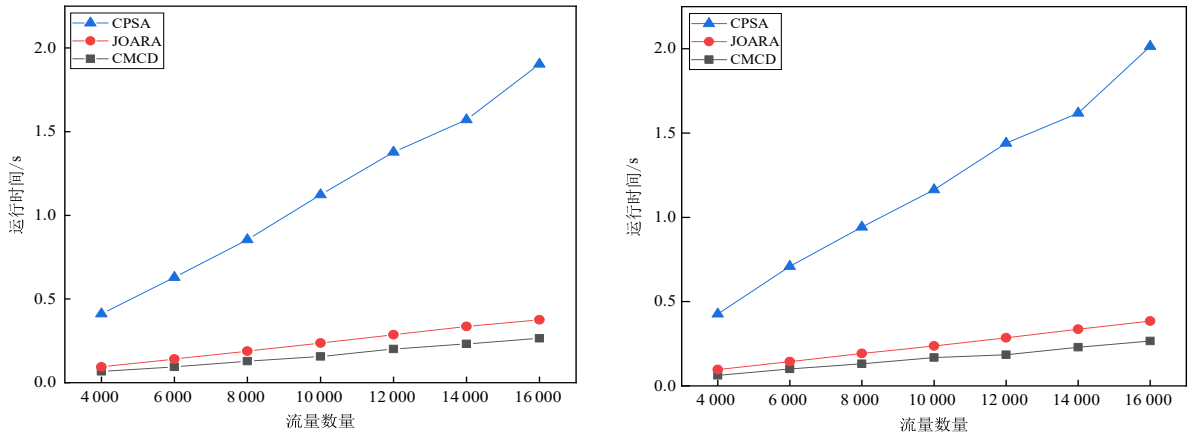
容量. JOARA 算法在不同流量下的吞吐量表现明显优于对比算法,并且随着流量增加更接近链路容量. 相比之下,CPSA 算法的吞吐量在 3 种算法中表现最差. 综合比较 3 种算法,在 Fat-Tree 拓扑中,JOARA 算法的吞吐量相较于 CMCD 算法和 CPSA 算法,分别提高了 13.86% 和 29.82%;在 ARPANet 拓扑中,JOARA 算法的吞吐量相较于 CMCD 算法和 CPSA 算法,分别提高了 15.10% 和 27.95%. JOARA 算法通过合理规划多种网络模态流量的路由,实现了控制器负载均衡的目标,提升了系统吞吐量,提高了网络的整体性能和稳定性.

本文针对流量数量从 4 000~16 000 条分别在两个网络拓扑中进行多次实验,观察流量数量对算法运行效率的影响,实验结果如图 5 所示. 从运行时间结果来看,本文算法 JOARA 的运行时间显著低于对比算法 CPSA,略高于对比算法 CMCD,其原因在于:CPSA 在每条流量的决策过程中需遍历候选路径并逐一评估控制器容量与链路约束,属于典型的贪心+容量约束建模,计算复杂度随流量数呈线性上升,因此运行时间显著更长;而 CMCD 算法仅在代价函数上进行单次优化和在线更新,复杂度较低,运行时间相对较短. 相比之下,本文提出的 JOARA 虽然在时间开销上略高于 CMCD,但通过联合建模与对偶更新机制,在吞吐量与控制器负载均衡方面均显著优于两类对比算法,体现了性能与开销间的合理权衡.



(a) Fat-Tree 拓扑　　(b) ARPANet 拓扑

图 4　网络系统吞吐量 vs 流量数量



(a) Fat-Tree 拓扑　　(b) ARPANet 拓扑

图 5　运行时间 vs 流量数量

6　结论

本研究针对多模态网络中分布式控制平面负载不均衡问题,提出了一种多模态流量路由与多模态网元-控制器分配的联合优化方法. 该方法通过先行分析历史流量数据,确定多模态网元与控制器的匹配关系,以实现负载均衡;随后设计考虑控制器资源约束的在线流量路由算法,动态调整流量路径以最大化系统吞吐量. 通过在 Fat-Tree 和 ARPANet 两个典型网络拓扑中

的对比实验,系统分析了路由与多模态网元-控制器分配联合优化算法在4 000~16 000条流量时的表现,还对比了不同控制器数量下的表现,验证了本文所提算法在吞吐量和负载均衡方面的显著优势.

参考文献

[1] KHALID S, MAHBOOB A, AZIM C F, et al. IDHOCNET: A novel ID centric architecture for ad hoc networks[J]. Journal of Computer Networks and Communications, 2016, 2016: 6438584.

[2] AHLGREN B, DANNEWITZ C, IMBRENDA C, et al. A survey of information-centric networking[J]. IEEE Communications Magazine, 2012, 50(7): 26-36.

[3] VENKATARAMANI A, KUROSE J F, RAYCHAUDHURI D, et al. MobilityFirst: A mobility-centric and trustworthy Internet architecture[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 74-80.

[4] 李军飞, 胡宇翔, 伊鹏, 等. 面向2035的多模态智慧网络技术发展路线图[J]. 中国工程科学, 2020, 22(3): 141-147.
LI J F, HU Y X, YI P, et al. Development roadmap of polymorphic intelligence network technology toward 2035[J]. Strategic Study of CAE, 2020, 22(3): 141-147. (in Chinese)

[5] 李挥, 邬江兴, 邢凯轩, 等. 多边共管的多模态网络标识域名生成管理解析原型系统[J]. 中国科学: 信息科学, 2019, 49(9): 1186-1204.
LI H, WU J X, XING K X, et al. Prototype and testing report of a multi-identifier system for reconfigurable network architecture under co-governing[J]. Scientia Sinica (Informationis), 2019, 49(9): 1186-1204. (in Chinese)

[6] 胡宇翔, 伊鹏, 孙鹏浩, 等. 全维可定义的多模态智慧网络体系研究[J]. 通信学报, 2019, 40(8): 1-12.
HU Y X, YI P, SUN P H, et al. Research on the full-dimensional defined polymorphic smart network[J]. Journal on Communications, 2019, 40(8): 1-12. (in Chinese)

[7] 涂化清, 廖君虎, 朱俊, 等. 多模态网络环境下网络模态共存与优化部署方法[J]. 电子学报, 2025, 53(5): 1650-1660.
TU H Q, LIAO J H, ZHU J, et al. Network modal coexistence and optimal deployment method in polymorphic network environment[J]. Acta Electronica Sinica, 2025, 53(5): 1650-1660. (in Chinese)

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

[9] LARA A, KOLASANI A, RAMAMURTHY B. Network innovation using OpenFlow: A survey[J]. IEEE Communications Surveys & Tutorials, 2014, 16(1): 493-512.

[10] WU J X, LI J F, SUN P H, et al. Theoretical framework for a polymorphic network environment[J]. Engineering, 2024, 39: 222-234.

[11] WU J X. On the revolution of the information network development paradigm[J]. Science China Information Sciences, 2022, 65(11): 213301.

[12] 胡宇翔, 崔子熙, 李子勇, 等. 基于领域专用软硬件协同的多模态网络环境构造技术[J]. 通信学报, 2022, 43(4): 3-13.
HU Y X, CUI Z X, LI Z Y, et al. Construction technologies of polymorphic network environment based on codesign of domain-specific software/hardware[J]. Journal on Communications, 2022, 43(4): 3-13. (in Chinese)

[13] HOU X D, GAO S, LIU N C, et al. L3Geocast: Enabling P4-based customizable network-layer geocast at the network edge[J]. IEEE Transactions on Mobile Computing, 2024, 23(8): 8323-8340.

[14] 刘泽英, 胡宇翔, 崔鹏帅, 等. 基于P4的转发与加密一体化技术[J]. 信息工程大学学报, 2023, 24(6): 734-740.
LIU Z Y, HU Y X, CUI P S, et al. Integrated technology of forwarding and encryption based on P4[J]. Journal of Information Engineering University, 2023, 24(6): 734-740. (in Chinese)

[15] HU Y X, LI D, SUN P H, et al. Polymorphic smart network: An open, flexible and universal architecture for future heterogeneous networks[J]. IEEE Transactions on Network Science and Engineering, 2020, 7(4): 2515-2525.

[16] 崔子熙, 田乐, 崔鹏帅, 等. 支持增量式编程的多模态网络环境[J]. 电子学报, 2024, 52(4): 1230-1238.
CUI Z X, TIAN L, CUI P S, et al. Enabling incremental programming in PINet environment[J]. Acta Electronica Sinica, 2024, 52(4): 1230-1238. (in Chinese)

[17] 李炯, 胡宇翔, 崔鹏帅, 等. 面向多模态网络环境的网络模态增量式部署机制研究[J]. 电信科学, 2023, 39(6): 33-43.
LI J, HU Y X, CUI P S, et al. Research on incremental deployment mechanism of network modality for polymorphic network environment[J]. Telecommunications Science, 2023, 39(6): 33-43. (in Chinese)

[18] 张汝云, 肖戈扬, 单麒赫, 等. 多模态网络下多智能体协

同控制的通信拓扑重构方法[J]. 通信学报, 2022, 43(4): 50-59.
ZHANG R Y, XIAO G Y, SHAN Q H, et al. Communication topology reconstruction method for multi-agent cooperative control in polymorphic networks[J]. Journal on Communications, 2022, 43(4): 50-59. (in Chinese)

[19] KAZEMIESFEH M, IMANPOUR S, MONTAZEROLGHAEM A. Enhanced load balancing technique for SDN controllers: A multi-threshold approach with migration of switches[J]. Computer Communications, 2025, 238: 108167.

[20] LI C L, LIU J, MA N, et al. Deep reinforcement learning based controller placement and optimal edge selection in SDN-based multi-access edge computing environments[J]. Journal of Parallel and Distributed Computing, 2024, 193: 104948.

[21] ZILBERMAN A, HADDAD Y, ERLICH S, et al. Heterogeneous SDN controller placement problem: The Wi-Fi and 4G LTE-U case[J]. Computer Networks, 2021, 198: 108376.

[22] NGUYEN D T, PHAM C, NGUYEN K K, et al. Jointly optimized resource allocation for SDN control and forwarding planes in edge-cloud SDN-based networks[J]. Future Generation Computer Systems, 2023, 145: 176-188.

[23] NASERI A, AHMADI M, POURKARIMI L. Placement of SDN controllers based on network setup cost and latency of control packets[J]. Computer Communications, 2023, 208: 15-28.

[24] AGUIRRE SANCHEZ L P, SHEN Y, GUO M Y. MDQ: A QoS-congestion aware deep reinforcement learning approach for multi-path routing in SDN[J]. Journal of Network and Computer Applications, 2025, 235: 104082.

[25] EL KAMEL A. Using FlowVisor and evolutionary algorithms to improve the switch migration in SDN[J]. Journal of Network and Computer Applications, 2024, 222: 103807.

[26] KABIRI Z, BAREKATAIN B, AVOKH A. GOP-SDN: An enhanced load balancing method based on genetic and optimized particle swarm optimization algorithm in distributed SDNs[J]. Wireless Networks, 2022, 28(6): 2533-2552.

[27] NOTO M, SATO H. A method for the shortest path search by extended Dijkstra algorithm[C]//Smc 2000 Conference Proceedings. 2000 IEEE International Conference on Systems, Man and Cybernetics. Cybernetics Evolving to Systems, Humans, Organizations, and Their Complex Interactions. Piscataway: IEEE, 2002: 2316-2320.

[28] AKYOL OZER E, SARAC T. MIP models and a matheuristic algorithm for an identical parallel machine scheduling problem under multiple copies of shared resources constraints[J]. Transactions in Operations Research, 2019, 27(1): 94-124.

[29] LEUNG J Y. Handbook of Scheduling: Algorithms, Models, and Performance Analysis[M]. Boca Raton: Chapman & Hall/CRC, 2004.

[30] BORODIN A, EL-YANIV R. Online Computation and Competitive Analysis[M]. Cambridge: Cambridge University Press, 2005.

[31] TU H Q, ZHAO G M, XU H L, et al. Tenant-grained request scheduling in software-defined cloud computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(12): 4654-4671.

[32] AL-FARES M, LOUKISSAS A, VAHDAT A. A scalable, commodity data center network architecture[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(4): 63-74.

[33] DING W L, XU H. Dynamic learning-based link restoration in traffic engineering with archie[C]//IEEE INFOCOM 2024 - IEEE Conference on Computer Communications. Piscataway: IEEE, 2024: 2428-2437.

[34] TU H Q, ZHAO G M, XU H L, et al. A robustness-aware real-time SFC routing update scheme in multi-tenant clouds[J]. IEEE/ACM Transactions on Networking, 2022, 30(3): 1230-1244.

[35] KIM D, LIU Z X, ZHU Y B, et al. TEA: Enabling state-intensive network functions on programmable switches[C]// Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication. New York: ACM, 2020: 90-106.

[36] GUO J M, YANG L, RINCÓN D, et al. Static placement and dynamic assignment of SDN controllers in LEO satellite networks[J]. IEEE Transactions on Network and Service Management, 2022, 19(4): 4975-4988.

[37] KOUTSOPOULOS I. Learning the jointly optimal routing and controller placement policy in mobile software-defined networks[C]//MILCOM 2023 - 2023 IEEE Military Communications Conference. Piscataway: IEEE, 2023: 484-490.

作者简介

涂化清　女,1995年4月出生于重庆市.现为浙江工商大学助理研究员.主要研究方向为软件定义网络、可编程数据平面、多模态网络等.
E-mail: thq@mail.zjgs.edu.cn

刘　硕　男,2001年8月出生于山西省晋中市.现为浙江工商大学信息与电子工程学院硕士研究生.主要研究方向为算力网络、多模态网络.
E-mail: 254754614@qq.com

方徐鑫　男,1999年12月出生于浙江省绍兴市.现为浙江工商大学信息与电子工程学院硕士研究生.主要研究方向为多模态网络、可编程网络.
E-mail: 22020090054@pop.zjgsu.edu.cn

马　博　男,1991年9月出生于宁夏回族自治区银川市.现为浙江工商大学信息与电子工程学院副教授、硕士生导师.主要研究方向为异构网优化、无人机通信、5G网络中的机器学习算法.
E-mail: mabo@mail.zjgsu.edu.cn

李传煌　男,1980年8月出生于江西省九江市.现为浙江工商大学信息与电子工程学院教授、硕士生导师.主要研究方向为新一代网络技术、算力网络、人工智能应用.中国电子学会会员编号:E190024269M.
E-mail: chuanhuang_li@zjgsu.edu.cn

朱　俊　男,1981年8月出生于浙江省嘉兴市.现为之江实验室工程专家、高级工程师.主要研究方向为新型网络体系结构、软件定义网络、网络资源管理、网络协议设计优化等.
E-mail: zhu_j@zhejianglab.org

邹　涛　男,1974年11月出生于重庆市.现为之江实验室研究专家、研究员.主要研究方向为多模态网络技术和新型网络体系架构.
E-mail: zout@zhejianglab.org