基于协同决策的 SDN 交换机迁移方案

熊 兵[1] 夏红芳[1] 赵锦元[2] 张 锦[1] 赵宝康[3] 李克勤[4]

[1](长沙理工大学计算机学院 长沙 410114)

[2](长沙师范学院信息科学与工程学院 长沙 410199)

[3](国防科技大学计算机学院 长沙 410073)

[4](纽约州立大学新帕尔兹分校计算机科学系 纽约 12561 美国)

摘 要 在软件定义网络(Software-Defined Networking,SDN)中,为解决控制器负载随网络流量波动而变化引起控制平面负载不均衡,甚至控制器过载的问题,本文提出一种基于协同决策的 SDN 交换机迁移方案,在保证控制平面负载均衡的同时,尽可能降低控制平面和数据平面之间的平均链路距离,以提升 SDN 网络性能。该方案首先在全面感知 SDN 网络运行状态的基础上,以需迁出交换机的过载控制器为中心,将邻近可用的控制器作为候选迁移目标,进而组成协同决策域。然后,综合考虑候选目标控制器的负载接收能力、过载控制器管理的交换机负载水平以及候选目标控制器和交换机之间的链路距离,逐步协商选出最佳交换机-控制器组合,最终形成 SDN 交换机迁移方案。最后,实验模拟仿真典型的 SDN 网络拓扑,对本文所提的 SDN 交换机迁移方案进行性能验证。实验结果表明:本文所提方案的过载率、平均链路距离和负载失衡因子比现有最优方案分别降低 78.1%、8%和 27.4%,可显著提升控制平面的可靠性和健壮性,优化 SDN 网络整体性能。

关键词 软件定义网络;交换机迁移;协同决策;平均链路距离;负载均衡程度

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2025.01601

SDN Switch Migration Scheme Based on Collaborative Decision Making

XIONG Bing[1] XIA Hong-Fang[1] ZHAO Jin-Yuan[2] ZHANG Jin[1] ZHAO Bao-Kang[3] LI Ke-Qin[4]

[1](School of Computer Science and Technology, Changsha University of Science and Technology, Changsha 410114)

[2](School of Information Science and Engineering, Changsha Normal University, Changsha 410199)

[3](School of Computer Science, National University of Defense Technology, Changsha 410073)

[4](Department of Computer Science, State University of New York at New Paltz, New York 12561 USA)

Abstract In the novel paradigm of Software-Defined Networking (SDN), the dynamic variability of network traffic will lead to the fluctuation of the load that switches bring to their controllers. Meanwhile, network topology sometimes will be changed by sudden events, such as link failures, switch faults, and other unexpected situations. Any topology change will sharply increase the load of network topology management in related controllers and the state synchronization between them and other controllers. These factors easily result in the load imbalance of SDN control plane, and even the problem of controller overload. Therefore, it is necessary to formulate a switch migration scheme before SDN deployments, to ensure stable and reliable network operations. However, existing switch migration schemes mainly focus on the load balance of SDN control plane, which result in strict migration conditions and ignore the impact of the distance

收稿日期:2024-08-27;在线发布日期:2025-03-31。本课题得到国家自然科学基金联合项目(U22B2005)、国家自然科学基金面上项目(61972412)、湖南省自然科学基金面上项目(2023JJ30053)、湖南省教育厅科研重点项目(22A0232,23A0735)资助。**熊 兵**,博士,副教授,主要研究领域为未来网络、网络安全和人工智能应用。E-mail:xiongbing@csust.edu.cn。**夏红芳**,硕士研究生,主要研究方向为 SDN 网络建模与优化。**赵锦元**,博士,副教授,主要研究领域为未来网络、网络测量、网络建模与优化。**张 锦**(通信作者),博士,教授,主要研究领域为软件工程、人工智能。E-mail:jzhang@csust.edu.cn。**赵宝康**(通信作者),博士,副教授,主要研究领域为计算机网络的系统设计、协议、算法和安全。E-mail:bkzhao@nudt.edu.cn。**李克勤**,博士,教授,主要研究领域为云计算、移动边缘计算、高性能计算、信息物理系统、计算机网络。

between migrated switches and their controllers on SDN network performance. To alleviate these problems, this paper proposes a SDN switch migration scheme based on collaborative decision making, for minimizing average link distances between SDN control plane and data plane, while achieving load balance of SDN control plane. This scheme first constructs a collaborative decision-making domain, based on the comprehensive perception of SDN network operation state. The domain is composed of all available controllers in the neighborhood of the overloaded controller whose switches need to be partly migrated out. Then, each controller within the domain is considered as a candidate migration target. For each candidate migration target, we select an optimal switch in the network domain administrated by the overloaded controller, as its candidate migration object. The selection comprehensively considers the load acceptance capacity of the candidate migration target, the load level of switches managed by the overloaded controller, and the link distances between the switches and the candidate migration target. By this way, we further build a set of candidate migration pairs with each target controller and its optimal switch for migration. Subsequently, we select the nearest migration pair from the candidate set and place it into the switch-controller pair set to be migrated. We update the load of the target controller, the minimum migration load and the switch set of the overloaded controller, and the candidate migration pair set. Afterward, we continuously select the best switch-controller migration pairs, to form the ultimate switch-controller pair set to be migrated, by repeatly performing the above process. Finally, we verify the performance of the SDN switch migration scheme proposed in this paper, by experiments with popular SDN simulation platform. In particular, we simulate a typical SDN network topology on the platform, and dynamically varying controller load for each switch. Experimental results show that the proposed scheme reduces overload rate, average link distance, and load imbalance factor respectively by 78.1%, 8%, and 27.4%, compared to state-of-the-art schemes. This implies that the proposed scheme can significantly enhance the reliability and stability of SDN control plane while optimizing the overall performance of SDN networks. Furthermore, the proposed scheme is expected to demonstrate good adaptability under different network scale and scenarios, such as data-center networks, wide-area networks, and Internet of Things.

Keywords software-defined networking; switch migration; collaborative decision making; average link distance; load balancing degree

1 引 言

软件定义网络(Software-Defined Networking, SDN)将控制逻辑从数据转发设备中解耦出来构建逻辑上集中的控制平面[1],进而通过 P4 编程语言灵活定义底层网络设备的分组转发行为,显著提升了网络的灵活性、可管控性和可编程能力。目前,SDN 已广泛应用于广域网、数据中心网络等场景中。对于广域网场景,Google 公司利用 SDN 技术设计 B4 系统,高效连接全球数据中心[2],提高网络利用率,增强流量管理的可预测性[3],并显著降低运营成本,推动网络技术创新[2]。对于数据中心网络场景,IBM 提出云控制器和网络控制器相结合的 SDN 架构[4],及时获取全局信息,实现云网络管理[5]。

在 SDN 应用中,通常部署多个控制器,对网络分区域进行管理。然而,网络流量的突发性和波动性通常会使交换机给控制器带来的负载产生波动变化[6]。同时,链路失效、交换机故障等突发情况引起的拓扑结构变化将会使相关控制器的网络拓扑管理负载以及与其他控制器之间的状态同步负载骤增。这些因素很容易导致各个控制器的负载不均衡,甚至出现控制器过载现象[7]。当控制器处于过载状态时,控制器无法及时生成并下发流规则,对应的网络分组转发时延大大增加,从而严重影响网络的数据传输性能[8]。此时,过载控制器需迁出部分交换机,

以确保控制平面稳定运行。总之，需设计动态的交换机迁移方案，使网络保持良好稳定的性能。

目前为止，已有许多研究工作通过贪心思想优化、博弈论思想优化以及 AI 算法智能训练等方法设计交换机迁移方案，以解决负载不均衡的问题。最初，研究者们利用贪心思想选择过载控制器中延迟最大[9]、距离最远以及请求速率高[10]的交换机迁出。然而，在控制器平均负载值过高的情况下，迁出这类交换机可能会导致频繁的交换机迁移，从而造成迁移效果不理想。还有不少研究人员将过载控制器视为玩家，其邻近控制器视为游戏参与者，过载控制器所管辖的交换机被看作资源或者待分配的对象，通过博弈最大化各自的利益，实现控制器平面负载均衡[11-12]。此外，也有研究人员[13]考虑用 AI 算法解决，基于控制器的历史负载预测未来的负载作为控制器的负载，提前预知过载控制器，然后根据自身的负载选取要迁移的交换机集合，并根据控制器的剩余处理能力选择单个目标控制器作为迁移对象。但是，将交换机集合迁移至单个目标控制器，可能会导致目标控制器负载过重，甚至出现无法迁移的情况。此外，现有方案很少考虑迁移交换机与目标控制器之间的链路距离，可能导致交换机迁移后与其所属的控制器距离较远，信息交互时延较长，进而影响网络数据传输性能。

针对上述的这些问题，本文提出了一种基于协同决策的 SDN 交换机迁移方案(SDN Switch Migration Based on Collaborative Decision Making，SMCD)。首先将交换机迁移问题抽象成任务分配和调度这一经典问题，为优化设计交换机迁移方案提供新的解决思路。在此基础上，将控制平面与数据平面之间的平均链路距离和控制平面负载均衡度作为优化目标，构建 SDN 交换机迁移优化模型。进一步，设计算法求解该优化模型，得到最优交换机-控制器集合。最后，采用实验对本文所提的交换机迁移方案进行性能验证。具体来说，本文做出了如下贡献：

(1)将交换机迁移问题抽象成任务分配和调度这一经典问题，通过将控制器和交换机分别抽象为实体和任务，以实体的任务接受能力和任务与实体之间的距离为依据建立优化模型，为优化设计交换机迁移方案提供了新的解决思路。

(2)提出了一种交换机动态迁移方案，在全面感知网络运行状态的基础上，采用协同决策的思想，让过载控制器的邻近可用控制器作为候选迁移目标，进而根据候选目标控制器的负载接收能力、过载控制器的交换机负载水平以及两者之间的链路距离，逐步择优选取交换机-控制器组合，以实现健壮可靠的控制平面，并优化网络性能。

(3)除了控制平面的负载均衡度之外，本文所提出的交换机迁移方案重点考虑控制平面与数据平面之间的平均链路距离，即在挑选迁移交换机时优先选择与决策域内控制器距离更近的交换机，有效提高控制平面和数据平面的消息交互效率，从而提高网络性能。

2　相关工作

为改善 SDN 控制平面的可靠性[14-15]，许多研究人员对 SDN 控制器的交换机迁移方案展开深入的研究，现有的解决方案主要是以响应时间、网络利用率、资源利用率等为优化目标制定交换机迁移方案。

为解决多域 SDN 网络中控制器负载不均衡的问题，Zhang 等人[16]采用贪心算法根据迁移概率选择需要迁出的交换机，网络代价选定目标控制器，有效地实现了控制平面的负载均衡。但每次迁移都需要所有控制器参与计算，增大了网络开销。Ye 等人[17]将交换机作为资源消耗者、控制器作为资源提供者，构建 SDN 资源消耗模型，进而将交换机迁移问题转化成资源利用率最大化问题。但该模型并未考虑控制器之间的消息交换与协调代价，增加了控制器之间的通信开销。Cui 等人[9]利用控制器的响应时间判断其负载情况，对控制器状态进行有效判断。在此基础上，选择过载控制器中延迟最大的交换机迁出，平均延迟最小的控制器迁入。胡涛等人[10]选择迁出具有高请求速率且距离过载控制器最远的交换机，并根据代价选择目标控制器。但在控制平面负载量普遍比较高的情形中，迁出上述的交换机，将导致交换机迁移多次，进而降低网络整体性能。Liu 等人[18]从未过载控制器中选择剩余资源最大的控制器，迁移开销最小的交换机集合，进而组合成控制器-交换机迁移对集合，再通过拍卖机制不断选取收益最大的迁移对。该选取策略减少了额外迁移开销，提高了网络效率，但未考虑控制器与交换机之间的通信开销。Prajapati 等人[19]构建一种基于贪心策略的最优交换机迁移方案，当某个控制器的负载均方差与控制平面平均负载量之差大于阈值时，该控制器的负载将迁移到未过载控制器上，以实现控制平面的负载均衡。但该方案仅关注负载均衡问题，而忽略了迁移效率问题。Lan 等人[20]指出迁

移效率的重要性,进而提出效率感知的交换机迁移方案,在平衡控制器负载的同时,提高迁移效率,以避免高昂的迁移代价和不必要的控制开销。进一步,Wang 等人[21]设计一种基于贪心算法的高效交换机迁移方案,通过交换机迁移触发指标感知负载不平衡,建立迁移效率模型,并采用贪心法选取负载较小、效率较高、但延迟较大的交换机作为迁出交换机,同时选取能接受迁出交换机的控制器作为目标控制器,从而实现迁移代价和负载均衡率之间的权衡。

一些研究者提出借鉴博弈论的决策机制,将邻近、空闲的控制器作为玩家,满足条件的交换机作为商品,玩家相互博弈使得最后各个玩家的利益最大化,从而确定合适的迁出交换机与目标控制器。Chen 等人[11]首次利用博弈机制解决交换机迁移问题,提出基于零和博弈的决策机制,将过载控制器与未过载控制器组成博弈区,同时区内控制器作为玩家,迁移交换机作为商品进行互相博弈,根据迁移交换机与博弈区内控制器的跳数和利用率选择迁出交换机和迁入控制器。Wu 等人[22]针对负载均衡与最大化资源利用率问题,设计控制器负载差异矩阵,以检测控制器之间的负载不平衡,进而触发交换机迁移。在此基础上,利用控制器之间的非合作博弈机制确定迁移的交换机,以使总体收益最大化。但复杂的非合作博弈机制需消耗更多的计算资源和时间。姚蓝等人[12]设计一种基于联盟博弈的自适应交换机迁移机制,将控制器和交换机映射问题建模为组合优化问题,通过控制器之间相互博弈,制定全局优化的控制器交换机映射方案。上述方案仅使用流请求数量作为资源分配的依据,忽略了流特性。针对该问题,Li 等人[23]提出基于流特性的动态控制器关联机制,采用具有合作博弈的贪心集覆盖算法,优先选择资源消耗大的交换机,由交换机提出请求,主从控制器与其他所有控制器进行博弈,构建控制器和交换机之间的近似最优关联方案。然而,受贪心算法的特性和合作博弈策略的影响,某些控制器将关联过多的交换机,造成控制平面负载不均衡。Cheng 等人[24]主要解决带约束条件的资源(CPU、带宽、内存)利用率最大化问题,并将该问题等价成非合作博弈理论中的最大化玩家利益问题。

随着人工智能的发展,不少研究者将 AI 技术引入到交换机迁移方案的设计中。Sun 等人[25]建立了一种基于多智能体强化学习的动态控制器负载均衡方案 MARVEL,主要实现动态控制器负载均衡,根据实时网络状况进行智能决策,提高控制平面的请求处理能力。Min 等人[26]利用 Q-learning 算法的自学习特性,提出一种面向可扩展 SDN 控制平面的动态交换机迁移算法,动态获取交换机迁移动作之间的时间间隔。该方案不仅考虑控制器的负载情况,还利用在线 Q-learning 实时调节迁移策略,不断优化控制器的负载分配。但是,该算法的效果较大程度上依赖学习率等参数的设置,不同参数的选择会影响算法的稳定性和执行效率。Yeo 等人[27]设计基于强化学习框架的交换机迁移方案,将交换机视为强化学习代理,考虑控制器和交换机的资源利用率,优化交换机迁移决策,提高控制平面的负载均衡度。但该方案根据实时负载数据迁移交换机,没有预测未来负载,将会造成目标控制器过载、迁移效率低等问题。Zhong 等人[13]基于控制器的历史负载量,利用深度学习预测未来负载量,并根据控制器负载选取迁移交换机集合,让过载控制器的负载降低至可承受水平。然而,该方案将交换机集合迁移至单个目标控制器,可能会导致交换机无法迁移或者控制平面负载失衡。Liu 等人[28]设计基于负载预测的双向交换机迁移方案,利用 ATT-GRU 模型预测控制器负载量,并基于改进的灰狼优化算法,实现多控制器之间的双向交换机迁移,有效缓解目标控制器过载问题。

尽管上述研究在一定程度上解决了负载均衡和迁移效率的问题,但普遍存在以下不足:(1)未充分利用不同控制器的剩余资源,导致在控制器负载普遍接近过载的情况下,无法生成有效的交换机迁移方案;(2)没有充分考虑控制器与交换机之间的链路距离对 SDN 的影响,导致交换机迁移后的网络性能明显下降;(3)部分方案的计算复杂度较大,也没有预测未来可能出现过载的情况,从而无法适应高动态网络场景。针对上述问题,本文在全面感知 SDN 网络运行状态的基础上,将过载控制器附近的所有可用控制器构成协同决策域,共同协商制定交换机迁移方案。在制定方案时,综合考虑交换机与目标控制器之间的链路距离,以及目标控制器接管交换机后的负载量与控制平面平均负载量的偏差两方面,优化选取交换机-控制器组合。通过该方案,期望在实现控制平面负载较为均衡的基础上,提高控制平面的总体运行效率,以实现健壮的控制平面[29]。

3　问题描述

在 SDN 网络中，控制器主要负责管理网络域中的交换机，并制定路由和转发决策，同时和其他控制器保持状态同步。当网络流量发生抖动，尤其是出现激增时，交换机将发送大量的流安装请求给控制器生成流规则，导致控制器无法及时响应处理，从而出现过载的情况。

对此，目前的交换机迁移方案通常考虑负载均衡度[30]、迁移代价[19]以及控制器到交换机之间的信息交互时延[31]等因素，进而设计优化算法选取目标控制器和迁移交换机。对于目标控制器，现有方案主要考虑所有控制器的剩余资源量，优先选择当前负载最小的控制器[9]。这样虽然能使控制器之间取得良好的负载均衡效果，但由于没有考虑迁移交换机与目标控制器的链路距离，导致交换机迁移后与控制器的信息交互时延长，进而影响其分组转发性能。对于迁移交换机，现有方案主要选取过载控制器中负载最大的交换机[13]，但可能找不到可接收的目标控制器。即使能找到目标控制器，但可能造成控制器之间负载失衡，且迁移交换机距离目标控制器远，信息交互慢。

图 1 给出了一个典型的控制器过载场景。对于过载控制器 c_2，按照现方案会将负载最大的交换机 s_4 迁移到负载最小的控制器 c_3，但这样会导致控制器 c_3 的负载过大，甚至无法接收。此情形下，应该将交换机 s_3 和 s_6 分别迁移到邻近可接收的控制器 c_1 和 c_3，既可以使得控制器之间的负载相对均衡，还可以使迁移交换机与控制器的链路距离更近。

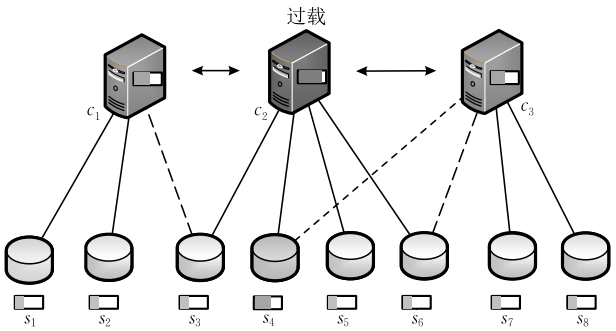


图 1　控制器过载场景

4　交换机迁移模型

4.1　设计思想

综上所述，现有交换机迁移方案主要考虑控制平面的负载均衡度，兼顾迁移代价、目标控制器到所选交换机的往返时延等因素。然而，交换机迁移后与目标控制器的链路距离对其信息交互时延，尤其是流安装时延有着显著的影响，进而严重影响其分组转发性能，最终对 SDN 网络整体性能产生不容忽视的影响。因此，在制定交换机迁移方案时，应提前迁移过载控制器所管辖的交换机，防止控制器进入过载状态，并使控制平面保持良好的负载均衡效果。对于迁出交换机，在确保目标控制器可接管的前提下，优先考虑迁出交换机与目标控制器之间的链路距离，以保证 SDN 整体性能。

基于上述动机，本文将控制器过载场景下的交换机迁移问题抽象为图 2 所示的任务分配和调度问题。在图 2 中，可调度分配的任务表示过载控制器管辖的交换机，其中的数字表示交换机给控制器产生的负载量；可接收任务的实体表示负载能力足够的控制器，其中的数字表示每个控制器可接收的负载量；任务和实体之间的相对距离表示交换机与控制器之间的链路距离。当控制器即将进入过载状态时，其管辖的部分交换机需就近迁移到其他控制器，且迁移后所有控制器的负载尽可能均衡。对于图 2 所示的问题抽象，即需将部分任务调度分配给其他实体，且到实体的距离尽可能近，并保证实体接收任务后的负载趋于均衡。

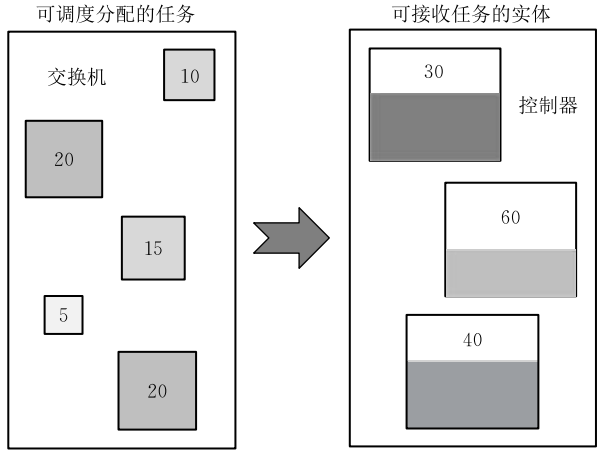


图 2　交换机迁移问题抽象

将交换机迁移问题抽象为任务分配和调度问题，不仅是理论上的简化，更是为解决复杂的 SDN 交换机迁移问题提供了一种通用框架。通过将控制器视为“任务接收实体”，交换机视为“需分配的任务”，可以借鉴任务调度领域的经典算法求解。此外，这一抽象也为交换机迁移方案的设计提供了明确的优化目标，即在负载趋于均衡的同时，尽可能降低控制平面与数据平面之间的平均链路距离，从而

提升 SDN 整体性能。

由于不同控制器对网络状态的感知范围有限，仅依靠单一控制器进行决策可能导致迁移方案缺乏全局视角，难以有效均衡控制平面负载，甚至无法制定出有效的交换机迁移方案。对此，本文引入协同决策思想，通过控制器之间实时共享网络状态信息建立全局网络视图，进而让邻近可用的控制器协商制定更优的交换机迁移方案。这种方式可有效提升迁移方案的准确性和稳定性，避免单一控制器决策带来的局部最优问题，从而更好地适应复杂的 SDN 网络环境。

基于上述思路，本文提出 SMCD 方案，首先通过网络域中的所有控制器实时共享网络状态信息，得到全局网络视图。然后采用负载预测算法提前获知所有控制器的负载状态，确定即将过载的控制器，其所管辖的所有交换机作为“可调度分配的任务”。对于需迁移交换机的控制器，附近可用的控制器构成“可接收任务的实体”。根据“可调度分配的任务”和“可接收任务的实体”，及两者之间的相对距离，不断优化选择交换机-控制器组合，最终形成交换机迁移方案。

4.2 概念定义

为便于描述 SMCD 方案，本文将 SDN 网络结构抽象为一个无向图 $G=(V,E)$，其中 V 表示 SDN 网络中的交换机与控制器集合，E 表示交换机和控制器之间的链路集合。不妨采用 $C=\{c_1,c_2,\cdots,c_N\}$ 表示控制器集合，$S=\{s_1,s_2,\cdots,s_M\}$ 表示交换机集合[29]。在 SDN 网络中，控制器与交换机往往是一对多的管理关系，因此控制器 c_i 与交换机 s_j 之间的管理关系可以形式化描述为

$$b_{ij}=\begin{cases}1, & c_i\text{管理}s_j\\ 0, & c_i\text{不管理}s_j\end{cases} \tag{1}$$

4.2.1 控制器负载

SDN 控制器负载主要由网络拓扑管理、该控制器所管理的所有交换机的交互响应以及该控制器与其他控制器的状态同步组成[32]。

网络拓扑管理负载[33]主要是源于控制器需要从其管辖的交换机定期收集信息，其中主要为链路信息。因此 c_i 的网络拓扑管理负载定义为

$$L_i^{\text{top}}=\sum_{j=1}^{N}b_{ij}h_{ij}f \tag{2}$$

其中 h_{ij} 为网络设备之间的跳数，f 为控制器收集链路状态信息的频率。

交换机交互响应负载本文主要是考虑控制器管辖的所有交换机发送给控制器的流安装请求[34]，以及该控制器和交换机的信息交互。根据上述内容，本文将 c_i 的交换机交互响应负载定义为

$$L_i^{\text{res}}=\sum_{j=1}^{M}b_{ij}\lambda_j+\sum_{j=1}^{M}b_{ij}\mu_{ij} \tag{3}$$

其中，λ_j 为 s_j 发送的 Packet-in 消息速率，μ_{ij} 为 c_i 与 s_j 的消息交换速率。

状态同步负载[35]主要源于控制器与网络域附近控制器定期状态同步，因此本文将 c_i 的状态同步负载定义为

$$L_i^{\text{syn}}=\sum_{c_i,c_k\in C}h_{ik}\eta \tag{4}$$

其中 η 为控制器之间的状态同步频率。

因此，任一控制器 c_i 的负载量定义为

$$L_i=L_i^{\text{top}}+L_i^{\text{res}}+L_i^{\text{syn}} \tag{5}$$

假设 c_i 能够处理的最大负载量为 $L_i^{\max}$，因此 c_i 的最小迁出负载量定义为

$$L_i^{\text{mig}}=L_i-\alpha L_i^{\max} \tag{6}$$

其中，α 为控制器过载系数，其取值范围通常设置为 $[85\%,95\%]$。

控制器 c_i 决定是否接受交换机 s_j 前，需要计算控制器的可接收负载量，主要综合考虑控制器能处理的最大负载量和控制平面的平均负载量两因素，其中控制平面的平均负载量 $\bar{L}$ 定义为

$$\bar{L}=\frac{1}{N}\sum_{i=1}^{N}L_i \tag{7}$$

假设 θ 为权重参数，c_i 期望达到的负载量定义为

$$L_i^{\text{exp}}=\theta\alpha L_i^{\max}+(1-\theta)\bar{L} \tag{8}$$

进一步，控制器 c_i 的可接收负载量定义为

$$L_i^{\text{rec}}=\begin{cases}L_i^{\text{exp}}-L_i, & L_i<L_i^{\text{exp}}\\ 0, & L_i\geq L_i^{\text{exp}}\end{cases} \tag{9}$$

控制平面的负载均衡程度采用负载失衡因子来度量，其定义为所有控制器负载标准差，如式(10)所示。负载失衡因子越小，控制平面的负载均衡程度越高。

$$\varphi=\frac{1}{N}\sqrt{\sum_{i=1}^{N}(L_i-\bar{L})^2} \tag{10}$$

4.2.2 平均链路距离

平均链路距离很大程度上决定了控制平面与数据平面之间的交互性能，直接影响控制器对交换机的响应时间，尤其是流安装时延，进而影响 SDN 数据传输性能。假设交换机 s_j 和控制器 c_i 之间的最短链路距离为 d_{ij}，结合消息交换速率 μ_{ij} 和映射关系 b_{ij}，则控制平面与数据平面的平均链路距离可表示为

$$\bar{d}=\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}b_{ij}d_{ij}\mu_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{M}b_{ij}\mu_{ij}} \tag{11}$$

假设所有交换机与其控制器的消息交换速率相同，则控制平面与数据平面的平均链路距离可简化为

$$\bar{d}=\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}b_{ij}d_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{M}b_{ij}} \tag{12}$$

4.3　模型构建

任意一个控制器在即将进入过载状态之前，需要将其管辖的部分交换机迁出。此时，邻近可用的控制器构成协同决策域，不妨表示为 $G=\{c_1,c_2,\cdots,c_n\}$。在协同决策域 G 中，任意一个控制器 c_i 满足条件：(1) 可接收负载量 L_i^{rec} 大于 0；(2) 与迁出控制器 c_k 的链路距离 d_{ik} 不超过预设的最远距离 D。若协同决策域中所有控制器的负载接收总量大于迁出控制器 c_k 的最小迁出量，即 $L_k^{\mathrm{mig}}\leq\sum_{i=1}^{|G|}L_i^{\mathrm{rec}}$，则可尝试制定交换机迁移方案。在此基础上，本文以控制平面的负载均衡度及其与数据平面的平均链路距离为优化目标，构建式(13)所示的 SDN 交换机迁移优化模型。其中，约束条件为每个交换机被且仅被一个控制器管辖。

$$\begin{aligned}&\min\ \varphi\&\bar{d}\\&\text{s.t. }\ b_{ij}\in\{0,1\},\forall i,j\\&\qquad\sum_{i=1}^{N}b_{ij}=1,\forall j\end{aligned} \tag{13}$$

4.4　求解算法

4.4.1　总体算法流程

为求解上述优化模型，本文采用协同决策的思想，设计交换机迁移方案，基本算法思路如下：(1) 预测控制器负载量，并获取控制器状态，将预期过载的控制器作为源控制器；(2) 找出源控制器附近可用的控制器作为目标控制器，从而构建为协同决策域；(3) 综合考虑控制平面负载量，协同决策域中控制器的负载接收能力，和过载的控制器管理的交换机之间的链路距离，共同协商选择交换机-控制器组合，组成候选迁移集合；(4) 在候选迁移集合中，选取控制器能接收且距离近的交换机进行迁移，并保证迁移后控制器负载不会明显超出控制平面的平均负载；(5) 反复执行步骤(3)和(4)，直至过载控制器预期能保持正常的负载水平为止。

本文所提交换机迁移方案 SMCD 的实现流程如算法 1 所示。首先，每个控制器 c_i 采用诸如 LSTM 的循环神经网络模型[13]，根据自身历史负载量预测下个时间段的负载量，并与过载阈值进行比较(第 1～4 行)。若预测负载量超出阈值，则该控制器需迁出交换机，于是向邻近控制器发送交换机迁移请求，构建协同决策域，进而协商制定交换机迁移方案，并执行交换机迁移过程(第 5～10 行)。否则向管理员发出告警信息，要求新增控制器(第 11～14 行)。若预测负载量未超出阈值，则该控制器可接收交换机。当收到交换机迁移请求时，计算自身的可接收负载量，进而回复响应消息，以加入协同决策域(第 15～23 行)。

算法 1.　基于协同决策的交换机迁移方案 SMCD

输入：网络拓扑 $G(V,E)$，当前控制器 c_{cur} 附近的控制器集合 $C[N']$(N' 表示 c_{cur} 附近的控制器个数)，控制器 c_{cur} 管理的交换机集合 $S[m]$(m 表示 c_{cur} 管理的所有交换机个数)，控制器 c_{cur} 能够承受的最大负载量 $L_{cur}^{\max}$，控制平面的平均负载量 $\bar{L}$

1. $L_{cur}\leftarrow PredictLoad(c_{cur})$;
2. $L_{cur}^{\mathrm{mig}}\leftarrow L_{cur}-\alpha L_{cur}^{\max}$;
3. $cdd\leftarrow\varnothing$; //协同决策域集合初始化
4. $css\leftarrow\varnothing$; //待迁移交换机-控制器集合初始化
5. IF $L_{cur}^{\mathrm{mig}}>0$ THEN
6. $cdd\leftarrow BuildCDD(C,L_{cur}^{\mathrm{mig}})$;
7. IF cdd is not empty, THEN
8. $css\leftarrow GenerateCSS(cdd,S,L_{cur}^{\mathrm{mig}},\bar{L})$;
9. IF css is not empty, THEN
10. $ExecuteSwitchMigration(css)$;
11. ELSE
12. Send an alarm message to administrators for adding controllers;
13. END IF
14. END IF
15. ELSE IF receiving a switch migration request from a controller c THEN
16. $L_{cur}^{\exp}=\theta\alpha L_{cur}^{\max}+(1-\theta)\bar{L}$;
17. IF $L_{cur}^{\exp}>L_{cur}$ THEN
18. $L_{cur}^{\mathrm{rec}}\leftarrow L_{cur}^{\exp}-L_{cur}$;
19. Send an acceptance response with L_{cur}^{rec} to c;
20. ELSE
21. Send a reject response to c;
22. END IF
23. END IF

4.4.2 协同决策域构建

下面给出协同决策域的构建过程，如算法 2 所示。首先，源控制器向邻近控制器发送消息，以请求进行交换机迁移(第 1～6 行)。然后，源控制器等待接收响应，直到收到全部控制器的响应或超时为止，接受请求的所有控制器构成协同决策域(第 7～19 行)。随后，源控制器对协同决策域中所有控制器的可接收负载量进行累加，得到可接受负载总量(第 20 行)。若该值不小于源控制的最小迁出量，则源控制器向接受请求的所有控制器发送协同决策域构建成功消息(第 21～24 行)。否则，发送协同决策域构建失败消息(第 25～31 行)。

算法 2. 协同决策域构建算法 BuildCDD

输入：控制器集合 $C[N']$，源控制器 sc 的最小迁移负载量 L_{sc}^{mig}

输出：协同决策域 cdd

1. $cdd \leftarrow \varnothing$;
2. FOR $i \leftarrow 1, N'$ DO
3. IF $d_{sc,i} \leq NCD\_MAX$ //两个邻近控制器之间的距离上界
4. send a switch migration request to $C[i]$;
5. END IF
6. END FOR
7. $n \leftarrow 0$; //协同决策域中的控制器数量初始化为 0
8. $start\_time \leftarrow GetSystemTime()$;
9. FOR $i \leftarrow 1, N'$ DO
10. $current\_time \leftarrow GetSystemTime()$;
11. IF $current_time - start_time >= $ TIMEOUT THEN
12. break;
13. END IF
14. receive a response r_i from a neighboring controller c;
15. IF r_i is an acceptance response
16. $cdd \leftarrow cdd \cup \{c\}$;
17. $n++$;
18. END IF
19. END WHILE
20. $L^{\text{total}} \leftarrow \sum_{i=1}^{n} L_i^{\text{rec}}$;
21. IF $L^{\text{total}} \geq L_{sc}^{\text{mig}}$ THEN
22. FOR $i \leftarrow 1, n$ DO
23. $SendControllerMessage(c_i,$ "The collaborative decision-making domain is successfully constructed");
24. END FOR
25. ELSE
26. FOR $i \leftarrow 1, n$ DO
27. $SendControllerMessage(c_i,$ "The collaborative decision-making domain is failed constructed");
28. END FOR
29. $cdd \leftarrow \varnothing$;
30. END IF
31. RETURN cdd;

4.4.3 交换机迁移方案制定

下面给出交换机迁移方案的制定过程，如算法 3 所示。首先，候选迁移对象为源控制器管辖的网络域中所选取的最佳交换机，候选迁移目标为协同决策域中的每个控制器(第 1～3 行)。然后，候选迁移目标和对应的最佳候选迁移对象组成候选迁移对，进而生成候选迁移集合(第 4～8 行)。在此基础上，选取距离最近的迁移对放入待迁移集合(第 10～11 行)，并更新其目标控制器的负载量，同时更新源控制器的最小迁出负载量及其管辖的交换机集合(第 12～14 行)。随后，更新候选迁移集合，即如果某个候选迁移对中的交换机和选定迁出的交换机相同，那么为其目标控制器再找一个源控制器管辖的最佳交换机(第 15～25 行)。反复执行上述候选迁移对选取与相关更新操作，一直选取到源控制器不再过载，或者候选迁移集合为空为止(第 9 行)。若候选迁移集合为空，则返回失败结果，否则返回待迁移集合(第 26～29 行)。

算法 3. 交换机迁移方案制定算法 GenerateCSS

输入：协同决策域 $cdd[n]$，源控制器 sc 所管理的交换机集合 $S_{sc}[m]$，源控制器 sc 的最小迁出负载量 L_{sc}^{mig}，控制平面的平均负载量 $\bar{L}$

输出：待迁移交换机-控制器集合 css

1. $CandidateCSS \leftarrow \varnothing$; //候选迁移集合初始化为空
2. FOR $i \leftarrow 1, n$ DO
3. $k \leftarrow FindBestSwitch(cdd[i], S_{sc}, \bar{L})$;
4. IF $k != 0$ THEN //没有找到合适的交换机
5. $CandidateCSS \leftarrow CandidateCSS \cup \{(cdd[i], S_{sc}[k])\}$;
6. END IF
7. END FOR
8. $css \leftarrow \varnothing$; //待迁移交换机-控制器集合初始化为空
9. WHILE $L_{sc}^{\text{mig}} > 0$ && $CandidateCSS$, THEN
10. $pair \leftarrow FindNearestPair(CandidateCSS)$;
11. $css \leftarrow css \cup pair$;
12. $L_{pair.c} \leftarrow L_{pair.c} + L_{pair.s}^{s}$; //$L_{pair.s}^{s}$ 表示交换机 $pair.s$ 产生的控制器负载量
13. $L_{sc}^{\text{mig}} \leftarrow L_{sc}^{\text{mig}} - L_{pair.s}^{s}$;
14. $DeleteSwitch(S_{sc}, pair.s)$;
15. FOR $i \leftarrow 1, CandidateCSS.size$ DO
16. IF $pair.s == CandidateCSS[i].s$
17. $k \leftarrow FindBestSwitch(CandidateCSS[i].c, S_{sc}, \bar{L})$;

```
18.        IF k!=0 THEN
19.          CandidateCSS[i].s←S_sc[k];
20.        ELSE
21.          DeleteCSPair(CandidateCSS[i]);
22.        END IF
23.      END IF
24.    END FOR
25. END WHILE
26. IF L_sc^mig>0 THEN
27.    css←∅;
28. END IF
29. RETURN css;
```

对于每个候选目标控制器，需要从源控制器管辖的所有交换机中挑选一个最佳的交换机作为候选迁移对象。最佳候选迁移交换机的搜索过程如算法 4 所示。依次遍历源控制器管辖的每个交换机(第 2 行)。首先，要求交换机的负载量不超过候选目标控制器的可接收负载量。同时，交换机迁移后目标控制器的负载量不超过控制平面平均负载量的 δ 倍，以保证控制平面取得良好的负载均衡(第 3 行)。然后，在满足上述要求的交换机中，选取离候选目标控制器最近的那个作为最佳候选迁移交换机(第 4～9 行)。

算法 4.　最佳候选迁移交换机搜索算法 FindBestSwitch

输入：候选目标控制器 c_i，源控制器 sc 所管理的交换机集合 $S_{sc}[m]$

输出：最佳候选迁移交换机编号 k

```
1. k←0; //第 0 个交换机实际上不存在，到任意控制器的距离默认为∞
2. FOR j←1,m DO
3.    IF L_j^s<L_i^rec && L_i+L_j^s<δL̄ THEN
          //L_j^s表示交换机 S_sc[j]产生的控制器负载量
4.      IF d_ij<d_ik THEN
5.        k←j;
6.      END IF
7.    END IF
8. END FOR
9. RETURN k;
```

4.4.4　交换机迁移方案执行

当交换机迁移方案制定完毕后，得到待迁移交换机-控制器集合。对于集合中的每对交换机-控制器，交换机迁移过程如图 3 所示。该过程大致分为四个阶段：(1)将目标控制器 c_k 修改为迁移交换机 s 的等效控制器；(2)源控制器 c_i 向迁移交换机 s 插入和删除空流表项，以确定迁移时机；(3)源控制器 c_i 使用 Barrier-request 消息刷新挂起请求；(4)将目标控制器 c_k 的角色由等效控制器变更为主控制器[36]。

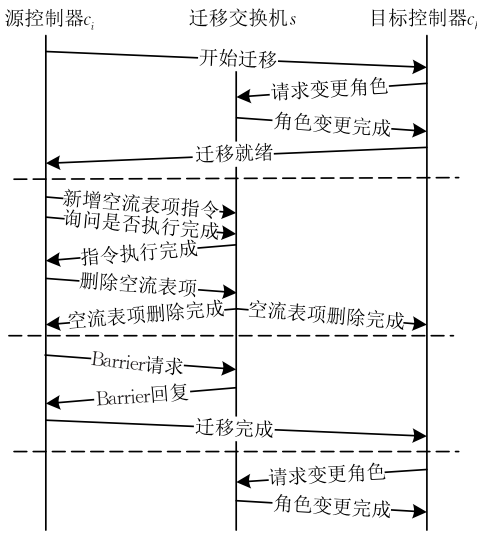


图 3　交换机迁移执行过程

4.4.5　时间复杂度分析

本文所提 SMCD 方案的时间复杂度可分解为上述四个阶段的算法复杂度。对于负载预测阶段，该方案采用 LSTM 负载预测模型，利用历史负载数据预测控制器的未来负载量。LSTM 模型分为训练和预测两个阶段，训练阶段离线完成，不影响方案在线运行；预测阶段的计算开销依赖于滑动窗口大小 W 和隐藏单元数量 H，时间复杂度为 $O(W\times H^2)$。对于协同决策域构建阶段，源控制器向邻近控制器发送交换机迁移请求并接收响应。假设源控制器有 N' 个邻近控制器，则协同决策域阶段的时间复杂度为 $O(N')$。

对于交换机迁移方案制定阶段，源控制器从其管辖的所有交换机中，为每个候选目标控制器选择最佳迁移交换机。假设源控制器管理的交换机个数为 m，协同决策域内控制器数量为 n，候选迁移对集合选取的时间复杂度为 $O(m\times n)$。随后，每选取一个交换机-控制器组合，需更新候选迁移对集合，即为新选定交换机涉及的迁移对重新选择最佳交换机。假设实际迁移的交换机个数为 l，每次需要更新候选迁移集合中的 k 个迁移对，迁移对更新需要遍历源控制器管理的 m 个交换机，则该阶段的时间复杂度为 $O(l\times k\times m)$。

综上所述，SMCD 交换机迁移方案的时间复杂度为 $O(W\times H^2)+O(N')+O(l\times k\times m)$。考虑到负载预测阶段中的滑动窗口大小 W 和隐藏单元数量 H 取值较小，所以 $O(W\times H^2)$ 可控，能满足在线需求。考虑到 SDN 网络域中每个控制器的邻近控制器个数 N' 往通常不多，因此 $O(N')$ 较低。考虑到每次实际迁移的交换机个数 l 往往只有几个甚至为

1，每次选择一个交换机-控制器组合对后需要更新的候选迁移对数量 k 同样较少甚至为 1，而源控制器管理的交换机数量 m 通常不大，因此 $O(l\times k\times m)$可控，同样可以满足在线需求。总之，本文所提 SMCD 方案实际执行的算法复杂度不高，可以满足 SDN 网络部署运行的实时性需求。

5 性能评估

5.1 实验方法

本文采用SDN仿真实验平台Mininet，选取RYU作为 SDN 控制器，验证本文所提 SMCD 方案的优越性。实验模拟一个典型的网络拓扑，共有 19 个节点和 25 条链路，每条链路设置一个距离值，如图 4 所示。该网络包含 15 台交换机（s_1，s_2，s_i，…，s_{15}），并划分为 4 个区域。每个区域部署一台控制器进行管理，共 4 台控制器（c_1、c_2、c_3、c_4）。

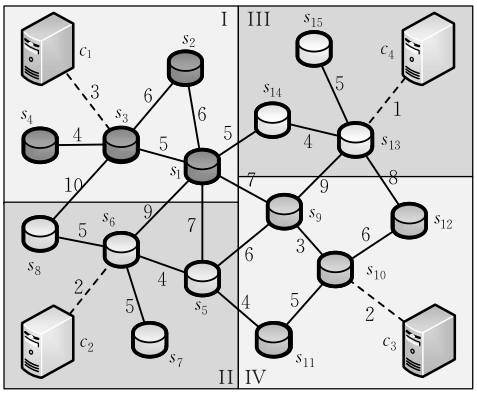
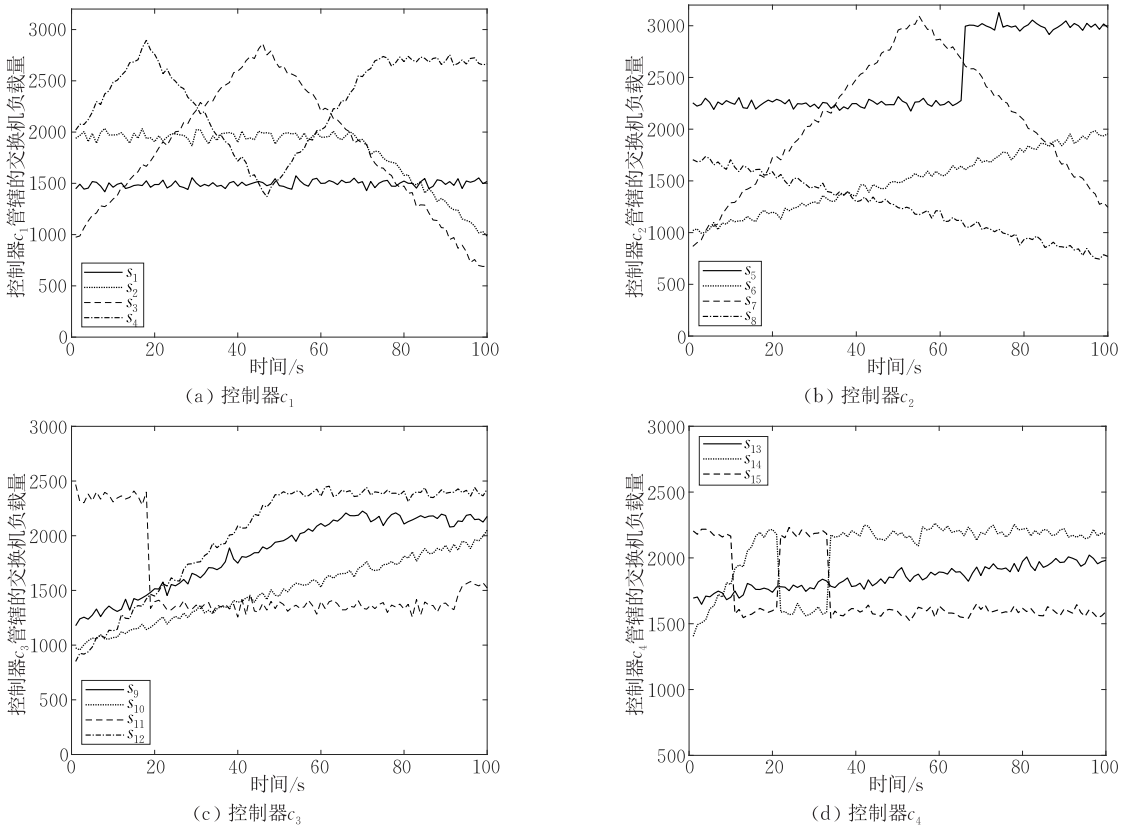


图 4 网络拓扑

基于上述网络拓扑，利用网络性能测量工具 Iperf，为每台交换机按不同变化规律生成动态流量，以模拟实际网络在不同条件下的流量变化情况，从而检验不同交换机迁移方案的运行性能。对于每个控制器管理的交换机，根据其动态流量和 packet-in 消息比例，可得其给控制器产生的负载量如图 5 所示。



（a）控制器c_1 （b）控制器c_2

（c）控制器c_3 （d）控制器c_4

图 5 不同控制器管辖的交换机产生的负载量

在图 5 中，不同控制器管辖的交换机产生的负载量呈现出不同的变化趋势，大致可分为六类：（1）负载量始终保持相对稳定，对应的交换机包括 s_1；（2）负载量稳步缓慢上升或下降，对应的交换机包括 s_6、s_8、s_{10}、s_{15}；（3）负载量只有前期稳步增加或后期稳定减少，其余时间则保持相对稳定，对应的交换机包括 s_2、s_9、s_{12}；（4）负载量始终稳步增加或减少，在达到极值点后反向变化或保持稳定，对应的交换机包括 s_3、s_4、s_7；（5）负载量除了在某些时刻急剧变化外，保持相对稳定，对应的交换机包括 s_5、s_{11}、s_{13}；（6）负载量前期稳步增加，除某些时刻变化外，保持相对稳定，对应的交换机包括 s_{14}。

假定每个控制器单位时间内能处理的最大负载量为 10 000，网络拓扑管理负载量为 800，与其他控制器进行状态同步的负载量为 500。结合图 4 给出的 SDN 网络拓扑结构和图 5 给出的每个交换机对控制器产生的负载量，可得每个控制器的总负载量如图 6 所示。控制器过载系数设置为 90%，即控制器的负载量达到 9000 则进入过载状态。从图 6 可以看出：控制器 c_1 的总负载量在前 17 s 稳步快速增长，并在第 15～71 s 内绝大部分时间处于过载状态，然后开始快速下降；控制器 c_2 的总负载量在前 55 s 稳步增长，并在第 46～83 s 处于过载状态，然后稳步下降；控制器 c_3 的总负载量在前 18 s 稳步增长，在 1 s 内急剧下降后稳步增长，并在第 83 s 后处于过载状态；控制器 c_4 的总负载量在前 9 s 稳步上升，然后急速下降并很快回升，之后一直保持相对稳定。

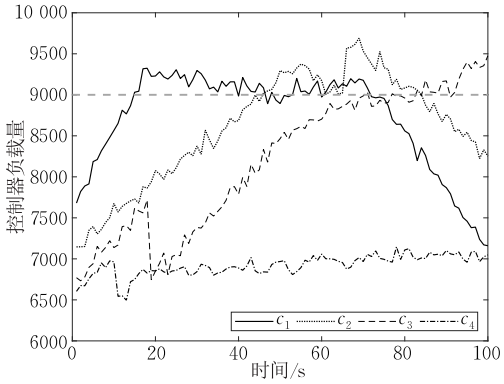
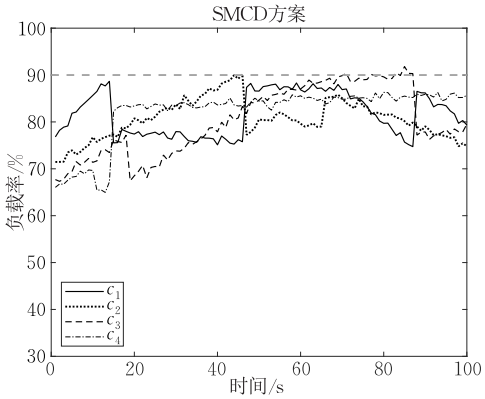


图 6　控制器的总负载量

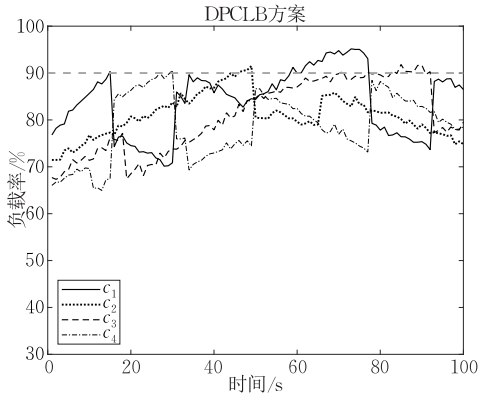
5.2　实验结果与分析

5.2.1　过载率

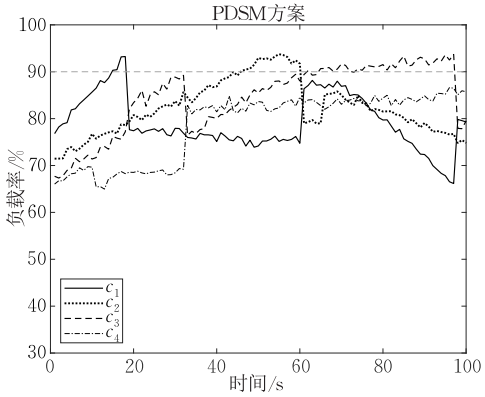
为有效衡量交换机迁移方案的效果，本文定义控制平面的过载率为其中有任一控制器处于过载状态的时间占总运行时间的比例，以反映 SDN 网络的可靠性和健壮性。对此，定义控制器在某个时刻的负载率为其当前负载量与最大处理能力的比值。当控制器的负载率达到 90% 时，即视其处于过载状态。在本实验中，控制器可接收负载量的权重参数 θ 设为 0.7，控制平面平均负载量的 δ 倍设为 1.2。基于上述 SDN 网络拓扑结构和交换机负载量，执行本文所提的 SMCD 方案和目前主流的 DPCLB 方案[10]、PDSM 方案[13]，并记录所有控制器的实时负载量，进而统计得到其负载率如图 7(a)～(c)所示。进一步，可计算得到上述三种方案以及无交换机迁移的过载率如图 7(d)所示。



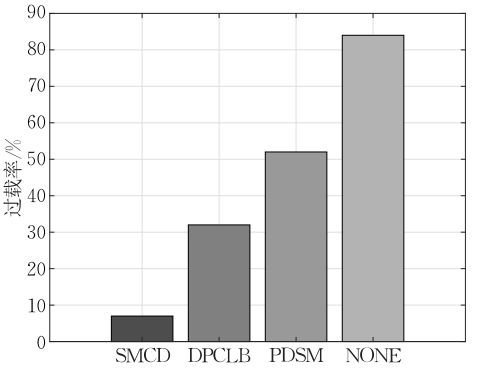
(a) SMCD方案的负载率



(b) DPCLB方案的负载率



(c) PDSM方案的负载率



(d) 不同交换机迁移方案的过载率

图 7　不同交换机迁移方案的负载率与过载率

从图 7 中可以看出:本文所提 SMCD 方案的控制平面过载率显著低于 DPCLB 方案和 PDSM 方案,分别降低了 78.1%和 86.5%。当无交换机迁移时,根据图 6 可计算得出,控制平面的过载率为 84%。对于本文所提 SMCD 方案,只有控制器 c_2、c_3 短时间出现过载,因而过载率仅有 7%。这主要归因于该方案根据协同决策域中控制器的可接收负载量和交换机产生的控制器负载量,选择交换机-控制器候选组合,进而尽可能迁移即将过载的控制器所管辖的交换机。对于 DPCLB 方案,控制器 c_1、c_2、c_3、c_4 都出现过载,其中控制器 c_1 过载状态的持续时间较长,过载率为 32%。这主要是因为该方案选择迁出具有高请求速率且距离过载控制器最远的交换机,导致控制器负载度高时交换机无法迁移。对于 PDSM 方案,控制器 c_1、c_2、c_3 较长时间处于过载状态,尤其是控制器 c_3 持续时间最长,过载率达到 52%。这主要是因为该方案要求迁移交换机的负载量小于目标控制器处理能力的 10%,而控制器负载度都比较高时没有合适的目标控制器可以接收,导致控制器长时间过载。

5.2.2 平均链路距离

控制平面与数据平面之间的平均链路距离对 SDN 网络性能有着关键性的影响,是衡量交换机迁移方案的主要性能指标之一。基于上述 SDN 网络拓扑结构和交换机负载量和参数配置,实验执行目前主流的 DPCLB 方案[10]、PDSM 方案[13]和本文所提的 SMCD 方案,统计得到其实时的平均链路距离如图 8 所示。

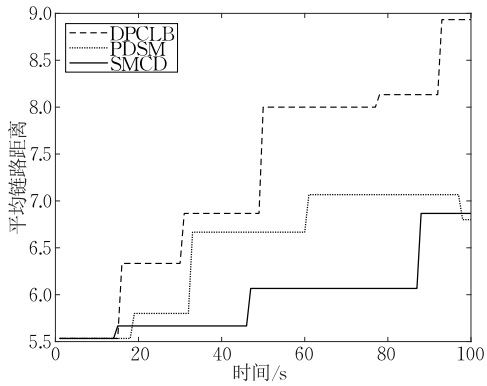


图 8 不同交换机迁移方案的平均链路距离

从图 8 中可以看出:本文所提 SMCD 方案的平均链路距离绝大部分时候都低于 DPCLB 和 PDSM 方案,其均值分别降低了 17.8%和 8%。具体来看,这三种方案的平均链路距离在初始阶段(14 s)保持在相同的最低水平。这是因为 SDN 控制器部署时使平均链路距离最小,尚未出现控制器过载问题。然后,本文所提 SMCD 方案、DPCLB 方案和 PDSM 方案的平均链路距离分别在第 14 s、第 15 s 和第 18 s 时开始增加。这主要是因为控制器 c_1 在第 15 s 开始进入过载状态,DPCLB 方案迁移交换机 s_3,而本文所提 SMCD 方案预测控制器负载,提前一个周期迁移交换机 s_1。PDSM 方案同样预测控制器负载,但要求迁移交换机负载小于目标控制器处理能力的 10%,导致延后迁移交换机。从第 19 s 到第 98 s,本文所提 SMCD 方案的平均链路距离一直低于 DPCLB 方案和 PDSM 方案。这是因为本文所提 SMCD 方案优先选择链路距离最短的交换机-控制器组合进行迁移,而 DPCLB 方案选择请求速率高的交换机迁出,PDSM 方案选择负载量最大的交换机迁出,对控制器与交换机的链路距离考虑少。

5.2.3 负载均衡度

对于 SDN 控制平面,负载均衡度对其总体运行性能有着显著影响,是衡量交换机迁移方案的关键性能指标之一。实验采用式(10)所示的负载失衡因子 φ 度量 SDN 控制平面的负载均衡度。基于上述 SDN 网络拓扑结构、交换机负载量和参数配置,实验执行目前主流的 DPCLB 方案[10]、PDSM 方案[13]和本文所提的 SMCD 方案,统计得到其实时的负载失衡因子如图 9 所示。

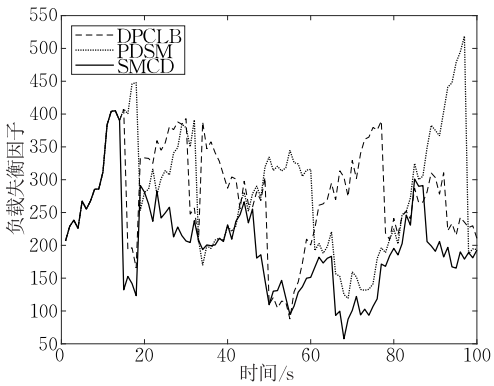


图 9 不同交换机迁移方案的负载失衡因子

从图 9 中可以看出:本文所提 SMCD 方案的负载失衡因子大部分时间明显低于 DPCLB 和 PDSM 方案,其均值分别降低了 27.4%和 28.1%。这是因为本文所提 SMCD 方案根据控制器的可接收负载量和控制平面的平均负载量选择接收交换机,使得控制器接收交换机后自身负载量更靠近控制平面的平均负载量。而 PDSM 方案选择负载加权和最大

的交换机迁移到自身剩余处理能力最大的控制器上，DPCLB 方案选择距离最远同时请求率最大的交换机迁移到选取代价最大的控制器上，对控制平面的平均负载量考虑少。以第 19 s 到第 46 s 为例，PDSM 方案的负载失衡因子初始缓慢增加，第 32 s 后急速下降，第 34 s 后稳步增加。这主要是因为第 32 s 时控制器 c_3 过载，将选择负载量最大的交换机 s_{11} 迁移到处理能力最大的控制器 c_4 上。同时，DPCLB 方案的负载失衡因子第 30 s 时开始下降，第 33 s 时激增而后开始下降。这主要是因为迁移时选择距离最远同时请求率最大的交换机 s_{14} 迁移到控制器 c_1 上；第 30 s 时，控制器 c_2 过载，DPCLB 方案需要再一次进行交换机迁移。与此形成对照的是，本文所提 SMCD 方案的负载失衡因子保持相对稳定。这是因为本文所提 SMCD 方案期间没有控制器过载，无需再次迁移。

5.2.4　SMCD 方案的适用性分析

上述实验结果表明：对于本文模拟的小型 SDN，SMCD 方案在平均链路距离和负载均衡度方面均有明显的优势。同时，该方案能够快速构建协同决策域，进而优化生成交换机-控制器组合，有效提高交换机迁移方案的制定成功率和优化水平。对于大规模 SDN，由于控制器和交换机数量众多，SMCD 方案在确定源控制器之后，更容易找到邻近控制器构建协同决策域，进而为过载控制器管辖的交换机找到更近的域内控制器接管，从而缩短迁移交换机与目标控制器的链路距离，进一步提高控制平面与数据平面之间的交互效率。

对于不同类型的网络环境，SMCD 方案具有良好的适用性。在软件定义数据中心网络中，以典型拓扑结构 Fat-Tree 为例，控制器往往通过核心交换机接入，每台汇聚交换机连接多台核心交换机，而每台边缘交换机到所有控制器的链路距离相近。根据 SMCD 方案的优化目标，交换机迁移后控制平面与数据平面的平均链路距离不会明显加长，甚至基本保持不变，有效保证了网络性能。在软件定义广域网中，控制器与交换机的地理分布范围广泛，导致控制器与交换机的链路距离往往较长。SMCD 方案优先选择链路距离较近的目标控制器接管交换机，从而尽可能减少控制平面与数据平面之间的平均链路距离，增强了 SDN 对广域网环境的适应能力。在软件定义物联网中，控制器通常部署在资源受限的边缘或中继节点上，计算能力有限。SMCD 方案采用协同决策的思想，充分利用所有邻近控制器的可用资源，大大提高了交换机迁移方案的制定成功率。

此外，对于流量波动大的网络环境，SMCD 方案能够通过实时感知网络状态，提前预测控制器的负载变化，并制定合理的交换机迁移方案，确保网络的稳定性和高效性。总之，SMCD 方案能够较好适应不同规模和类型的 SDN 网络场景。

6　结　论

在 SDN 网络部署中，现有交换机迁移方案的迁移条件严格，且很少考虑控制平面与数据平面之间的平均链路距离。对此，本文将交换机迁移问题抽象成任务分配和调度问题，将控制器和交换机分别抽象为实体和任务，以实体的任务接受能力和任务与实体之间的距离为依据，制定任务优化分配方案，为交换机迁移问题提供新的解决思路。进一步，本文提出一种基于协同决策的 SDN 交换机迁移方案 SMCD。该方案在全面感知网络运行状态的基础上，让过载控制器附近的可用控制器组成协同决策域。在此基础上，综合考虑交换机负载水平、候选目标控制器的负载接收能力、控制平面的平均负载量以及交换机与目标控制器之间的链路距离，逐步产生最优交换机-控制器组合，最终形成交换机迁移方案。

实验借助 SDN 仿真实验平台，模拟典型的 SDN 网络拓扑结构，并让每个交换机模拟产生动态变化的控制器负载量，以评估本文所提 SMCD 方案的性能。实验结果表明：本文所提 SMCD 方案的主要性能指标均优于 DPCLB 方案和 PDSM 方案。首先，本文所提 SMCD 方案的过载率为 7%，比 DPCLB 方案和 PDSM 方案分别降低了 78.1% 和 86.5%。其次，本文所提 SMCD 方案的平均链路距离比 DPCLB 方案和 PDSM 方案分别降低了 17.8% 和 7.7%。最后，本文所提 SMCD 方案的负载失衡因子比 DPCLB 方案和 PDSM 方案分别降低了 27.4% 和 28.1%。总之，本文所提 SMCD 方案显著提升了 SDN 控制平面的可靠性和健壮性，优化了 SDN 网络整体性能。

在未来的研究工作中，针对不同类型不同规模的网络场景，比如数据中心网络、广域网和物联网中，验证本文所提 SMCD 方案在不同网络拓扑结构和控制器负载变化规律下的可靠性、健壮性和适用

性。同时,将搭建真实的 SDN 网络实验平台,通过注入真实网络流量,测试和评估 SDN 交换机迁移方案的实际应用效果。此外,针对现有的负载预测模型,将扩展其训练场景,比如轻载休眠、故障失效,以增加其泛化能力,为设计更通用可靠的交换机迁移方案提供支持。最后,针对控制器故障失效和轻载休眠的场景,在现有控制器故障检测与发现机制,以及负载预测机制的基础上,设计高效健壮的 SDN 交换机迁移方案,以确保 SDN 在各种网络场景下始终能高效稳定运行。

致 谢 长沙理工大学先进技术研究院 SDN 研究团队的同学们在 SDN 网络建模与优化方面积累的诸多经验,为本文的完成与撰写提供了极大的帮助。感谢国家自然科学基金、湖南省自然科学基金、湖南省教育厅科研项目的资助,在此由衷地表示感谢,同时感谢在百忙之中评阅的各位专家!

参 考 文 献

[1] Shen G, Li Q, Shi W, et al. Modeling and optimization of the data plane in the SDN-based DCN by queuing theory. Journal of Network and Computer Applications, 2022, 207, 103481: 1-14

[2] Hong C Y, Mandal S, Al-Fares M, et al. B4 and after: Managing hierarchy, partitioning, and asymmetry for availability and scale in Google's software-defined WAN// Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM). Budapest, Hungary, 2018: 74-87

[3] Jain S, Kumar A, Mandal S, et al. B4: Experience with a globally-deployed software defined WAN. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 3-14

[4] Zhang Z, Li H, Dong S, et al. Software defined networking (SDN) research review//International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE). Qingdao, China, 2018: 291-300

[5] Banikazemi M, Olshefski D, Shaikh A, et al. Meridian: An SDN platform for cloud network services. IEEE Communications Magazine, 2013, 51(2): 120-127

[6] Chien W C, Lai C F, Cho H H, et al. A SDN-SFC-based service-oriented load balancing for the IoT applications. Journal of Network and Computer Applications, 2018, 114: 88-97

[7] Mokhtar H, Di X, Zhou Y, et al. Multiple-level threshold load balancing in distributed SDN controllers. Computer Networks, 2021, 198, 108369: 1-16

[8] Zhang Shao-Jun, Lan Ju-Long, Hu Yu-Xiang, et al. Survey on scalability of control plane in software-defined networking. Journal of Software, 2018, 29(1): 160-175(in Chinese)
(张少军, 兰巨龙, 胡宇翔等. 软件定义网络控制平面可扩展性研究进展. 软件学报, 2018, 29(1): 160-175)

[9] Cui J, Lu Q, Zhong H, et al. A load-balancing mechanism for distributed SDN control plane using response time. IEEE Transactions on Network and Service Management, 2018, 15(4): 1197-1206

[10] Hu Tao, Zhang Jian-Hui, Wu Jiang, et al. Controller load balancing mechanism based on distributed policy in SDN. Acta Electronica Sinica, 2018, 46(10): 2316-2324 (in Chinese)
(胡涛, 张建辉, 邬江等. SDN 中基于分布式决策的控制器负载均衡机制. 电子学报, 2018, 46(10): 2316-2324)

[11] Chen H, Cheng G, Wang Z. A game-theoretic approach to elastic control in software-defined networking. China Communications, 2016, 13(5): 103-109

[12] Yao Lan, Lan Ju-Long. Adaptive SDN switch migration mechanism based on coalitional game. Journal on Communications, 2020, 41(8): 1-10(in Chinese)
(姚蓝, 兰巨龙. 基于联盟博弈的自适应 SDN 交换机迁移机制. 通信学报, 2020, 41(8): 1-10)

[13] Zhong H, Xu J, Cui J, et al. Prediction-based dual-weight switch migration scheme for SDN load balancing. Computer Networks, 2022, 205,108749: 1-12

[14] Zhu L, Karim M M, Sharif K, et al. SDN controllers: A comprehensive analysis and performance evaluation study. ACM Computing Surveys, 2020, 53(6): 1-40

[15] Sufiev H, Haddad Y, Barenboim L, et al. Dynamic SDN controller load balancing. Future Internet, 2019, 11(3): 75-96

[16] Zhang J, Hu T, Zhao W, et al. DDS: Distributed decision strategy based on switch migration towards SDN control plane//International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery(CyberC). Nanjing, China, 2017: 486-493

[17] Ye X, Cheng G, Luo X. Maximizing SDN control resource utilization via switch migration. Computer Networks, 2017, 126: 69-80

[18] Liu Y, Gu H, Yan F, et al. Highly-efficient switch migration for controller load balancing in elastic optical inter-datacenter networks. IEEE Journal on Selected Areas in Communications, 2021, 39(9): 2748-2761

[19] Prajapati U, Bijoy C C, Amit B, et al. OptiGSM: Greedy-based load balancing with minimum switch migrations in software-defined networks. IEEE Transactions on Network and Service Management, 2023, 21(2): 2200-2210

[20] Hu T, Lan J, Zhang J, et al. EASM: Efficiency-aware switch migration for balancing controller loads in software-defined networking. Peer-to-Peer Networking and Applications, 2019, 12(2): 452-464

[21] Wang C, Hu B, Chen S, et al. A switch migration-based decision-making scheme for balancing load in SDN. IEEE Access, 2017, 5: 4537-4544

[22] Wu G, Wang J, Obaidat M S, et al. Dynamic switch migration with noncooperative game towards control plane scalability in SDN. International Journal of Communication Systems, 2019, 32(7): 1-13

[23] Li Z, Hu Y, Hu T, et al. Dynamic SDN controller association mechanism based on flow characteristics. IEEE Access, 2019, 7: 92661-92671

[24] Cheng G Z, Chen H C, Hu H C, et al. Dynamic switch migration towards a scalable SDN control plane. International Book Title of Communication Systems, 2016, 29(9): 1482-1499

[25] Sun P, Guo Z, Wang G, et al. MARVEL: Enabling controller load balancing in software-defined networks with multi-agent reinforcement learning. Computer Networks, 2020, 177, 107230: 1-10

[26] Min Z, Hua Q, Jihong Z. Dynamic switch migration algorithm with Q-learning towards scalable SDN control plane//9th International Conference on Wireless Communications and Signal Processing (WCSP). Nanjing, China, 2017: 1-4

[27] Yeo S, Naing Y, Kim T, et al. Achieving balanced load distribution with reinforcement learning-based switch migration in distributed SDN controllers. Electronics, 2021, 10(2): 162-178

[28] Liu Q, Liu Y, Meng Q, et al. BSM-LP: Bidirectional switch migration with controller load prediction for software-defined Internet of Things. IEEE Internet of Things Journal, 2024, 11(13): 24196-24209

[29] Xiong Bing, Xia Hong-Fang, Xu Shuai, et al. A SDN Switch Migration Scheme Based on Collaborative Decision Making. 202310669673.5, China, 2024.01.04(in Chinese)
(熊兵，夏红芳，许帅等. 一种基于协同决策的 SDN 交换机迁移方案. 202310669673.5，中国，2024.01.04)

[30] Wang Ying, Yu Jin-Ke, Pei Ke-Ke, et al. A load informing based load balancing mechanism for multiple controllers in SDN. Journal of Electronics & Information Technology, 2017, 39(11): 2733-2740(in Chinese)
(王颖，余金科，裴科科等. 基于负载通告的 SDN 多控制器负载均衡机制. 电子与信息学报，2017，39(11)：2733-2740)

[31] Zhang S J, Lan J L, Sun P H, et al. Online load balancing for distributed control plane in software-defined data center network. IEEE Access, 2018, 6: 18184-18191

[32] Yao G, Bi J, Li Y L, et al. On the capacitated controller placement problem in software defined networks. IEEE Communications Letters, 2014, 18(8): 1339-1342

[33] Metter C, Gebert S, Lange S, et al. Investigating the impact of network topology on the processing times of SDN controllers //IFIP/IEEE International Symposium on Integrated Network Management(IM). Ottawa, Canada, 2015: 1214-1219

[34] Sahoo K S, Puthal D, Tiwary M, et al. ESMLB: Efficient switch migration-based load balancing for multicontroller SDN in IoT. IEEE Internet of Things Journal, 2019, 7(7): 5852-5860

[35] Chen Y, Zhang N, Yang J. A survey of recent advances on stability analysis, state estimation and synchronization control for neural networks. Neurocomputing, 2023, 515: 26-36

[36] Dixit A A, Hao F, Mukherjee S, et al. ElastiCon: An elastic distributed SDN controller//10th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS). California, USA, 2014: 17-28

XIONG Bing, Ph. D., associate professor. His research interests include future networks, network security and artificial intelligence applications.

XIA Hong-Fang, M. S. candidate. Her main research interests focus on SDN modelling and optimization.

ZHAO Jin-Yuan, Ph. D., associate professor. Her main research interests include future network, network measurements, network modelling and optimization.

ZHANG Jin, Ph. D., professor. His research interests include software engineering, and artificial intelligence.

ZHAO Bao-Kang, Ph. D., associate professor. His research interests include system design, protocols, algorithms, and security issues in computer networks.

LI Ke-Qin, Ph. D., professor. His research interests include cloud computing, mobile edge computing, high performance computing, cyber-physical systems, and computer networks.

Background

Software-Defined Networking (SDN), as a novel network paradigm of decoupling logic control from data forwarding, significantly promotes network programmability, manageability and controllability. It is worth noting that, the dynamic variability of network traffic easily leads to the fluctuation of controller load, resulting in the load unbalance of SDN control plane, and even controller overload. Therefore, it is necessary to formulate a switch migration scheme before SDN deployments, for stable and reliable network operations. However, existing switch migration schemes mainly focus on the load balance of control plane, which result in strict migration conditions and ignore the impact of the distance between migrated switches and their controllers on SDN network performance.

To alleviate these problems, we propose a SDN switch migration scheme based on collaborative decision making, for minimizing average link distances between SDN control plane and data plane while ensuring the load balance of SDN control plane. The main contributions of this paper include: (1) abstracting the problem of switch migration as task allocation and scheduling, to match switches under the administration of an expected overloaded controller with neighboring controllers that can accept load, which provides a new solution to the problem; (2) proposing a SDN switch migration scheme based on collaborative decision making, which gradually selects the best switch-controller migration pairs to achieve a robust and reliable control plane and optimize network performance; (3) our proposed switch migration scheme focuses on average link distance between control plane and data plane, besides of the load balance of control plane, which improves the message exchange speed between the two planes and further optimizes SDN network performance.

This work was supported in part by the National Natural Science Foundation of China (No. U22B2005, No. 61972412), the Hunan Province Natural Science Foundation of China (No. 2023JJ30053) and the Scientific Research Project of the Hunan Province Education Department (No. 22A0232, No. 23A0735).