Cooperative Route Management for Profit-Oriented Flows in Multi-Domain SDN Networks

You-Chiun Wang, *Senior Member, IEEE,* and Meng-Yu Chou

Abstract—This paper investigates SDN for route management in multi-domain networks, where each domain is independently controlled and inter-domain cooperation is required for cross-domain routing. To capture traffic heterogeneity, each flow is associated with a profit. We propose CRM-PF (Cooperative Route Management for Profit-oriented Flows), a framework that jointly maximizes *overall achieved profit (OAP)* and minimizes *packet loss rate (PLR)*. In CRM-PF, controllers perform intra-domain routing, coordinate cross-domain paths, and reroute flows under congestion. Link bandwidth is allocated based on flow category, unit profit, and demand, with a Nash bargaining game to resolve bandwidth contention on borrowed links. Simulation results show that CRM-PF improves throughput, reduces PLR, and increases OAP over existing methods, demonstrating its effectiveness for profit-oriented routing in multi-domain SDN networks.

Index Terms—multi-domain network, Nash bargaining, profit, route management, software-defined networking (SDN).

1 Introduction

The rapid growth of network traffic and the increasing heterogeneity of applications have made efficient network management increasingly challenging. Traditional networks tightly couple the control and data planes within switches, which limits flexibility and makes policy deployment complex and error-prone [1]. *Software-defined networking (SDN)* addresses this limitation by decoupling control and data planes and enabling centralized control and programmability, thereby simplifying network management and optimization [2].

In large-scale deployments such as campus and enterprise networks, SDN is typically organized into multiple administrative domains, each managed by an autonomous controller. This *multi-domain SDN (MDS)* architecture improves scalability and fault isolation, but also introduces a key challenge: efficient coordination across independently operated domains for end-to-end service provisioning.

As illustrated in Fig. 1, MDS networks require inter-domain cooperation to support cross-domain routing. While existing research on distributed SDN has largely focused on controller placement and load balancing [3], less attention has been given to route management under heterogeneous traffic demands and dynamic congestion.

In practical networks, flows are inherently heterogeneous. Some flows (e.g., video conferencing) are highly sensitive to delay and loss, while others (e.g., adaptive streaming) are more tolerant. To capture this difference in service requirements, we associate each flow with a *profit* that reflects its QoS demand, and define the achieved profit based on delivery performance.

This leads to a fundamental problem in MDS networks: *how to jointly optimize routing and bandwidth allocation across multiple autonomous domains under congestion while incorporating flow-level profit considerations.* The difficulty arises from the decentralized nature of domain control, which limits global coordination, and from the inherent trade-off between maximizing profit-driven utilization and preserving service quality under congestion.

The authors are with the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan (e-mail: ycwang@cse.nsysu.edu.tw; taylorchou6609@gmail.com).

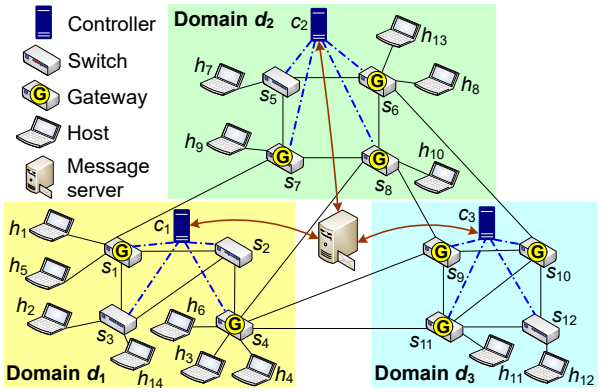


Fig. 1: An MDS network with three domains.

Existing SDN solutions, for example, ONOS SDN-IP, offer scalable inter-domain connectivity via BGP (border gateway protocol)-based mechanisms [4]. However, these solutions are primarily designed for reachability and interoperability across heterogeneous networks, and are not designed to support flow-level service differentiation and optimization objectives such as congestion-aware routing and profit-oriented bandwidth allocation.

Therefore, there remains a need for a practical framework that jointly addresses 1) flow-level service differentiation, 2) congestion-aware routing and bandwidth allocation, and 3) coordinated decision-making across multiple SDN domains.

Motivated by these observations, we propose a *cooperative route management framework for profit-oriented flows (CRM-PF)* for MDS networks, where the objective is to maximize *overall achieved profit (OAP)* while reducing *packet loss rate (PLR)*. CRM-PF enables each controller to perform local routing decisions while coordinating with neighboring domains for cross-domain path establishment. Under congestion, flows may be rerouted or allocated bandwidth based on their profit and demand, and inter-domain link borrowing is used to improve resource utilization when necessary.

Compared with ONOS SDN-IP, CRM-PF provides a more fine-

TABLE 1: Comparison of ONOS SDN-IP and CRM-PF.

aspect	ONOS SDN-IP	CRM-PF
design objective	IP reachability	profit maximization
routing granularity	prefix-based	flow-level
flow differentiation	none	profit + category
domain cooperation	implicit	explicit coordination
resource sharing	none	link borrowing
congestion handling	limited	rerouting + bandwidth control

grained and optimization-driven routing framework that explicitly considers flow profit and congestion dynamics, as summarized in Table 1. CRM-PF differs from ONOS SDN-IP in several key aspects, including design objective, routing granularity, flow differentiation, domain cooperation, resource sharing, and congestion handling. In particular, while ONOS SDN-IP focuses on IP reachability using prefix-based routing, CRM-PF operates at the flow level and enables profit-oriented and congestion-aware decision-making. Furthermore, CRM-PF supports explicit inter-domain coordination and link borrowing, which are not considered in ONOS SDN-IP, thereby improving overall network efficiency.

The main contributions of this paper are as follows:

- We design a profit-oriented route management framework for MDS networks that supports intra- and cross-domain flows, leveraging link borrowing to enhance routing flexibility and resource utilization under congestion.
- We propose differentiated congestion mitigation mechanisms to reroute flows or regulate bandwidth by category, and use a *Nash bargaining (NB)* game for compensation-based allocation of borrowed-link bandwidth.
- Extensive simulations under diverse congestion scenarios and demand-profit settings demonstrate that the proposed CRM-PF framework outperforms existing approaches in terms of throughput, average PLR, and OAP.

The remainder of this paper is organized as follows. Section 2 surveys the related work, and Section 3 presents the system model. Section 4 describes the CRM-PF framework, followed by a discussion of its challenges and improvements in Section 5. Performance evaluation results are reported in Section 6. Finally, Section 7 concludes the paper and outlines future work.

2 Related Work

2.1 Distributed SDN Control

A substantial body of research on distributed SDN control focuses on load balancing among controllers [5]–[8]. In these approaches, each controller manages a subset of switches, and overloaded controllers offload switches to others to decrease the control-plane load. This dynamic reassignment, referred to as *switch migration*, effectively balances controller workload. However, these studies primarily redistribute control responsibilities rather than optimize data-plane routing for flows.

The controller placement problem has also been extensively investigated. It affects multi-controller SDN performance in terms of implementation cost [9], fault tolerance [10], energy efficiency [11], and network resilience [12]. A link discovery mechanism is proposed in [13] to maintain topology information in multi-controller SDN environments. However, these studies do not consider flow routing or congestion mitigation, which are the focus of this work.

2.2 SDN-Based Route Management

Applying SDN to route management in data center networks has attracted attention. The study [14] adjusts routes to balance switch loads while reducing querying overhead. Fu *et al.* [15] use deep Q-learning to achieve high throughput for elephant flows and low latency for mice flows. The work [16] proposes a topology-aware routing scheme based on a regular topology description language. In [17], flows are decomposed and then forwarded along less congested paths. The study [18] increases received profit and reduces packet loss, while [19] considers priorities, energy consumption, and path loads. Liu *et al.* [20] design a multipath routing method using deep reinforcement learning. Despite their effectiveness, most approaches rely on well-structured topologies (e.g., fat-tree).

Moreover, route management under partial SDN deployment, where OpenFlow and legacy switches coexist, is studied in [21]. For hybrid SDNs with electrical and optical switches, differentiated QoS is achieved in [22] by regulating high-speed link usage based on flow ranks. Energy efficiency and load balancing are addressed in [23], while long-term revenue maximization via route selection is considered in [24]. Multipath routing using disjoint paths with near-optimal traffic splitting is investigated in [25]. However, these studies generally assume a single-controller architecture, limiting their applicability to multi-domain or distributed SDN environments.

2.3 MDS Route Management

Existing studies on MDS networks examine routing from multiple perspectives. Wibowo *et al.* [26] outline key challenges, including architecture, scalability, interoperability, and reliability, but do not address cross-domain cooperation. Collaborative multi-domain routing has been explored. For example, the study [27] proposes a framework that routes flows across domains while satisfying delay and bandwidth constraints and maximizing utilization via integer linear programming. However, it does not consider economic incentives for domain cooperation or differentiated congestion management.

Other studies focus on inter-controller cooperation. In [28], controllers coordinate to optimize cross-domain routing paths and maintain seamless IP services for mobile devices. Ibrahim *et al.* [29] examine joint flow routing and resource allocation to reduce energy consumption. Congestion has been addressed using link borrowing and priority-based bandwidth allocation [30], [31]. While these methods improve bandwidth utilization, they do not consider profit in routing decisions.

Several experimental studies have investigated hybrid MDS and legacy network deployment. Dawadi *et al.* [4] implement SDN-IP over ONOS for migrating legacy IPv4 networks to MDS-enabled IPv6 networks, evaluating controller placement and router migration. Thapa *et al.* [32] evaluate interoperability between legacy and SDN-IP networks in a Mininet testbed, showing improved bandwidth and latency and confirming SDN feasibility for ISP deployment. Despite their practicality, these studies do not address differentiated congestion mitigation.

In contrast, the proposed CRM-PF framework enables flow-level routing with a profit-oriented design. It addresses congestion via cross-domain coordination, adaptive link borrowing, and an NB game for bandwidth allocation, while incorporating profit-based payments. Table 2 compares CRM-PF with prior work, highlighting its ability to jointly support domain cooperation, congestion handling, and profit orientation.

3 System Model

Consider an MDS network composed of multiple domains. Each domain d_k contains a controller c_k, a set of OpenFlow-compliant switches, and attached hosts. Two domains, d_k and d_l, are *neighbors* if a switch in d_k connects to a switch in d_l; such switches

TABLE 2: Comparison of MDS routing studies and the proposed CRM-PF framework

study	scope	domain cooperation	congestion handling	profit orientation
Wibowo *et al.* [26]	MDS overview	not discussed		
Moufakir *et al.* [27]	multi-domain routing	explicit	partial	
Hata *et al.* [28]	seamless IP services	explicit		
Ibrahim *et al.* [29]	energy-aware routing	explicit		
Wang and Chang [30]	congestion mitigation	explicit	✓	
Wang and Yen [31]	congestion mitigation	explicit	✓	
Dawadi *et al.* [4]	hybrid MDS/legacy network	implicit		
Thapa *et al.* [32]	hybrid MDS/legacy network	implicit		
CRM-PF (this work)	flow-level routing	explicit	✓	✓

are known as *gateways*, and their controllers are also considered neighbors. Fig. 1 illustrates an example with three domains (d_1, d_2, d_3) managed by controllers (c_1, c_2, c_3). The gateway sets are $\{s_1, s_4\}$, $\{s_6, s_7, s_8\}$, and $\{s_9, s_{10}, s_{11}\}$, respectively. Each controller, switch, and host belongs to exactly one domain, ensuring non-overlapping domains.

A flow $f_{x,y}$ originates from source host h_x and terminates at destination host h_y. From the viewpoint of controller c_k, $f_{x,y}$ is a *domain-interested (DI)* flow if h_x or h_y resides in its domain d_k. For example, $f_{2,1}$ is a DI flow for c_1, and $f_{1,9}$ is a DI flow for both c_1 and c_2. If neither host belongs to d_k, $f_{x,y}$ is a *link-borrowing (LB)* flow. Let $\Phi(h_x)$ denote h_x's domain. There are two types of LB flows:

- **Type I:** $\Phi(h_x) \neq \Phi(h_y) \neq d_k$, $\Phi(h_x)$ and $\Phi(h_y)$ are not neighbors, and $f_{x,y}$ must traverse d_k. For instance, removing links (s_4, s_9) and (s_4, s_{11}) in Fig. 1 disconnects d_1 and d_3. Flow $f_{2,12}$ (from h_2 in d_1 to h_{12} in d_3) must pass d_2, making it a type-I LB flow from c_2's perspective.
- **Type II:** When the controller of $\Phi(h_x)$ or $\Phi(h_y)$ (not c_k) cannot identify a non-congested intra-domain route, it borrows links from d_k to reroute $f_{x,y}$. For example, flow $f_{1,4}$ has two candidate routes in d_1: $h_1 \to s_1 \to s_2 \to s_4 \to h_4$ and $h_1 \to s_1 \to s_3 \to s_2 \to s_4 \to h_4$. If link (s_2, s_4) is congested, both routes become infeasible. Hence, c_1 borrows links from d_2, yielding the route $h_1 \to s_1 \to s_7 \to s_8 \to s_4 \to h_4$. From c_2's perspective, $f_{1,4}$ is classified as a type-II LB flow.

Moreover, a message server facilitates inter-controller communication for establishing cross-domain routes, including LB flows and cross-domain DI flows (i.e., $\Phi(h_x) \neq \Phi(h_y)$). Controllers exchange only essential information without exposing domain internals (e.g., topology and non-gateway switches), while each controller independently determines intra-domain routing. Specifically, for an LB flow $f_{x,y}$ handled by c_k, the controller of $\Phi(h_x)$ (not c_k) cannot determine routing in d_k but is limited to providing auxiliary information. This design reduces message overhead while preserving domain autonomy.

Let $\mathcal{F}$ denote the set of all flows in the MDS network. Each flow $f_{x,y} \in \mathcal{F}$ is associated with a profit $\zeta_{x,y}$ defined as

$$\zeta_{x,y} = \zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y} \cdot t_{x,y}, \tag{1}$$

where $\zeta^{\mathrm{U}}_{x,y}$, $b_{x,y}$, and $t_{x,y}$ represent the unit profit, bandwidth demand, and flow duration of $f_{x,y}$, respectively. The unit profit captures the application type and its QoS requirement and is defined within $[\zeta_{\mathtt{min}}, \zeta_{\mathtt{max}}]$, where $\zeta_{\mathtt{min}}$ and $\zeta_{\mathtt{max}}$ are positive integers with $\zeta_{\mathtt{min}} < \zeta_{\mathtt{max}}$. A higher unit profit indicates stricter QoS requirements; for example, with $\zeta_{\mathtt{min}} = 1$ and $\zeta_{\mathtt{max}} = 10$, real-time, adaptive streaming, and non-real-time applications can be assigned unit profits of 10, 5, and 3, respectively. The formulation in Eq. (1) jointly considers QoS, bandwidth, and duration, thereby preventing starvation of bandwidth-intensive but QoS-insensitive flows (e.g., data backup).

Suppose that $f_{x,y}$ produces $n^{\mathtt{GEN}}_{x,y}$ packets in total, of which $n^{\mathtt{LOS}}_{x,y}$ packets are lost due to link congestion. Then, the problem can be formulated as follows:

$$\text{maximize} \quad \sum\nolimits_{f_{x,y} \in \mathcal{F}} \zeta_{x,y} \cdot \Theta(f_{x,y}), \tag{2}$$

$$\text{minimize} \quad \sum\nolimits_{f_{x,y} \in \mathcal{F}} \frac{n^{\mathtt{LOS}}_{x,y}}{n^{\mathtt{GEN}}_{x,y}} \cdot \frac{n^{\mathtt{GEN}}_{x,y}}{\sum_{f_{x',y'} \in \mathcal{F}} n^{\mathtt{GEN}}_{x',y'}}, \tag{3}$$

subject to

$$0 \leq n^{\mathtt{LOS}}_{x,y} \leq n^{\mathtt{GEN}}_{x,y} \qquad \forall f_{x,y} \in \mathcal{F}, \tag{4}$$

$$\zeta_{\mathtt{min}} \leq \zeta^{\mathrm{U}}_{x,y} \leq \zeta_{\mathtt{max}} \qquad \forall f_{x,y} \in \mathcal{F}, \tag{5}$$

$$\sum\nolimits_{f_{x,y} \in \mathcal{F}_\xi} b^{\xi}_{x,y} \leq \theta(\xi), \qquad \forall \xi \in \mathcal{L}, \tag{6}$$

$$0 \leq b^{k}_{x,y} \leq b_{x,y} \qquad \forall f_{x,y} \in \mathcal{F}, \forall d_k \in \mathcal{D}. \tag{7}$$

For objectives, Eq. (2) maximizes OAP, where $\Theta(f_{x,y}) \in [0, 1]$ denotes the *profit gain factor* of flow $f_{x,y}$. Eq. (3) minimizes the weighted average PLR across all flows, where the weight of each flow is proportional to its total generated packets. This weighting prevents flows with large traffic volumes from being underrepresented and avoids trivial solutions that prioritize only high-profit flows while starving others.

For constraints, Eq. (4) ensures that a flow cannot lose more packets than it generates. Eq. (5) imposes both lower and upper bounds on unit profit. In Eq. (6), $b^{\xi}_{x,y}$ denotes the bandwidth allocated to $f_{x,y}$ on link ξ, while $\theta(\xi)$ is the link capacity and $\mathcal{L}$ is the set of all links; this ensures that the total allocated bandwidth on each link does not exceed its capacity. In Eq. (7), $b^{k}_{x,y}$ is the bandwidth allocated to $f_{x,y}$ in domain d_k, and $\mathcal{D}$ is the set of all domains, ensuring that allocated bandwidth does not exceed flow demand.

Regarding the factor $\Theta(f_{x,y})$, we consider two formulations.

Option 1 (linear function): For a general case, the effective profit scales linearly with the packet delivery ratio:

$$\Theta(f_{x,y}) = 1 - n^{\mathtt{LOS}}_{x,y} / n^{\mathtt{GEN}}_{x,y}. \tag{8}$$

Option 2 (step function): For SVC (scalable video coding) flows, $\Theta(f_{x,y})$ is modeled as a non-decreasing step function:

$$\Theta(f_{x,y}) = \frac{\sum_{i=0}^{\beta-1} w_i \cdot \mathbf{1}(r_{x,y} \geq \tau_i)}{\sum_{i=0}^{\beta-1} w_i}, \tag{9}$$

where $r_{x,y}$ is the achieved transmission rate, β is the number of layers, τ_i is the minimum bandwidth requirement for the i-th layer, w_i is the corresponding weight, and $\mathbf{1}(\cdot)$ is the indicator function. This formulation captures the layered structure of SVC, where higher layers contribute only if all lower layers are delivered. The normalization ensures $\Theta(f_{x,y}) \in [0, 1]$, and typically $w_0 \geq w_1 \geq \cdots \geq w_{\beta-1}$.

We emphasize that traffic heterogeneity is captured in our model by assigning different unit profits to flows and adopting different functions for the profit gain factor. Table 3 summarizes acronyms and notations.

TABLE 3: Summary of acronyms and notations.

Acronyms

acronym	definition
BA	bandwidth adjustment
CFM	cooperative flow management
CRM-PF	cooperative route management for profit-oriented flows
DI/LB flow	domain-interested/link-borrowing flow
MCRM	multi-domain collaborative route management
MDS	multi-domain SDN (SDN: software-defined networking)
NB	Nash bargaining
OAP	overall achieved profit
PLR	packet loss rate

Notations

notation	definition
$\mathcal{F}$, $\mathcal{F}_\xi$	set of flows in the MDS network; flows on link ξ
d_k, $\Phi(h_x)$	domain of controller c_k; domain of host h_x
s_i	switch ($s_{\texttt{in}}$, $s_{\texttt{out}}$: ingress/egress gateways)
$f_{x,y}$	flow ($h_x \Rightarrow h_y$): unit profit $\zeta^{\texttt{U}}_{x,y}$, flow duration $t_{x,y}$, bandwidth demand $b_{x,y}$, profit gain factor $\Theta(f_{x,y})$
$b^l_{x,y}, b^\xi_{x,y}$	bandwidth of $f_{x,y}$ in domain d_l and on link ξ
$t^l_{x,y}$	total borrowing duration of $f_{x,y}$ in domain d_l
$t_{\texttt{bor}}$	duration of a single borrowing instance
$\theta(\xi), \theta_{\texttt{A}}(\xi)$	link capacity and residual bandwidth of ξ
σ	payment ratio
u_i, g_i	utility and disagreement point of player z_i

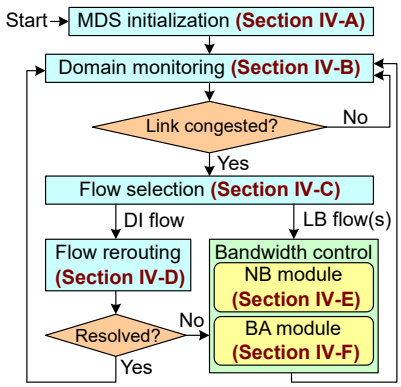


Fig. 2: Flowchart of the CRM-PF framework.

4 The Proposed CRM-PF Framework

Fig. 2 shows the flowchart of CRM-PF. The MDS network is first initialized (Section 4.1), and each controller monitors its domain status (Section 4.2). When a link becomes congested, a flow is selected (Section 4.3), and rerouting is applied to DI flows (Section 4.4). If congestion persists or all flows are LB flows, bandwidth control is performed by the NB module for type-II LB flows (Section 4.5) and the *bandwidth adjustment (BA)* module for the remaining flows (Section 4.6). Then, the design rationale is discussed in Section 4.7.

4.1 MDS Initialization

In domain d_k, each switch s_i discovers devices connected to its ports using the link layer discovery protocol. These devices may include 1) a host attached to s_i, 2) another switch within d_k, and 3) a gateway of an external domain (if s_i serves as a gateway). The switch reports the discovered information to its controller, c_k, enabling c_k to obtain the topology of d_k and identify the neighboring domains connected via its gateways.

Communication among controllers is facilitated through a publisher-subscriber architecture. A message server acts as a broker that relays messages from publishers to subscribers. Publishers send notifications to designated channels via the broker, while subscribers register to receive messages from these channels. Each controller may function as a publisher, a subscriber, or both. The ZeroMQ toolkit [33] can be used to implement this architecture.

TABLE 4: Five types of inter-controller messages.

message	parameters
domain_adv	$d_{\texttt{pub}}$, $\langle d_{\alpha_1}, m_{\alpha_1}, q_{\alpha_1}\rangle, \cdots, \langle d_{\alpha_a}, m_{\alpha_a}, q_{\alpha_a}\rangle$
cross_relay	$d_{\texttt{pub}}$, $\Phi(h_x)$, $\Phi(h_y)$, $s_{\texttt{in}}$, $s_{\texttt{out}}$, h_x, h_y, $b_{x,y}$, $\zeta^{\texttt{U}}_{x,y}$
ask_help	$d_{\texttt{pub}}$, $s_{\texttt{in}}$, $s_{\texttt{out}}$, h_x, h_y, $b_{x,y}$, $\zeta^{\texttt{U}}_{x,y}$, σ, $t_{\texttt{bor}}$
can_help	$d_{\texttt{pub}}$, $s_{\texttt{in}}$, $s_{\texttt{out}}$, h_x, h_y
rate_limit	$d_{\texttt{pub}}$, h_x, h_y, $b^l_{x,y}$

Table 4 presents inter-controller messages, the first two of which support cross-domain routing. Controller c_k exchanges *domain_adv* messages with neighbors to learn domain reachability. Each message contains the publisher domain ($d_{\texttt{pub}}$) and a list of domain-routing entries. Each entry $\langle d_{\alpha_j}, m_{\alpha_j}, q_{\alpha_j}\rangle$ indicates that the number of domains to cross from d_k to d_{α_j} is m_{α_j}, and q_{α_j} is a sequence number stamped by c_{α_j}.

Cross_relay messages are used to construct the route for a type-I LB flow $f_{x,y}$. Each message contains the following parameters: publisher domain ($d_{\texttt{pub}}$), source and destination domains ($\Phi(h_x)$, $\Phi(h_y)$), ingress and egress gateways within $d_{\texttt{pub}}$ ($s_{\texttt{in}}$, $s_{\texttt{out}}$), source and destination hosts (h_x, h_y), bandwidth demand ($b_{x,y}$), and unit profit ($\zeta^{\texttt{U}}_{x,y}$).

The remaining three message types handle type-II LB flows. When controller c_k intends to borrow links for rerouting $f_{x,y}$, it sends an *ask_help* message to neighboring controllers. The parameter σ denotes the *payment ratio*, determining the fraction of $f_{x,y}$'s profit used as compensation for link borrowing ($0 < \sigma < 1$; see Section 4.4). Moreover, $t_{\texttt{bor}}$ represents the expected duration of link borrowing.

When domain d_l can lend links, its controller c_l transmits a *can_help* message to c_k. Once the borrowed links in d_l later become congested, c_l issues a *rate_limit* message. Here, $b^l_{x,y}$ is the updated bandwidth allocated to $f_{x,y}$ by d_l, where $b^l_{x,y} < b_{x,y}$. The computation of $b^l_{x,y}$ depends on σ (see Section 4.5).

4.2 Domain Monitoring

Using the `EventOFPStateChange` procedure, controller c_k tracks the status of each switch s_i in domain d_k. Specifically, `MAIN_DISPATCHER` means a stable connection, while `DEAD_DISPATCHER` indicates a lost connection, prompting c_k to remove s_i from the topology of d_k. Once the connection is restored, s_i is re-added to the topology.

Similar to the destination-sequenced distance vector protocol, c_k can learn how to reach other domains by exchanging *domain_adv* messages with neighboring controllers. This information is maintained in its *domain-info* table. Each entry is of the form $\langle d_{\texttt{tar}}, d_{\texttt{neg}}, m_j, q_j\rangle$, where $d_{\texttt{tar}}$ denotes the target domain, $d_{\texttt{neg}}$ is the next-hop domain toward $d_{\texttt{tar}}$, m_j is the number of domains to traverse, and q_j is a sequence number.

For example, assume that d_1 and d_3 are not neighbors in Fig. 1, i.e., links (s_4, s_9) and (s_4, s_{11}) are removed. Then, the domain-info table of c_1 contains three entries: $\langle d_1, d_1, 0, q_1\rangle$, $\langle d_2, d_2, 1, q_2\rangle$, and $\langle d_3, d_2, 2, q_1\rangle$.

In addition, c_k queries each switch for statistics such as port capacity and the number of sent and received packets via the `OFPFlowStatsRequest` and `OFPPortStatsRequest` procedures. To keep an up-to-date view of d_k, c_k maintains three tables: the *host-info*, *route-info*, and *flow-info* tables.

When c_k identifies a host h_x, it adds an entry $\langle h_x, s_i, p_j\rangle$ to the host-info table, indicating that s_i is the switch to which h_x is connected via port p_j. Fig. 1 illustrates an example from the perspective of c_2. The corresponding entry for host h_7 is $\langle h_7, s_5, 3\rangle$, where h_7 is connected to switch s_5 via port 3.

TABLE 5: Example of the flow-info table (for controller c_2).

flow	attributes	s_5	s_6	s_7	s_8	PLR
$f_{7,10}$	(3, 5, 30)	3265	0	3241	2607	20.15%
$f_{11,13}$	(5, 8, 50)	0	3344	0	3880	13.81%

If h_x communicates with a host h_y in an external domain, c_k also records an entry $\langle h_y, s_{\texttt{out}}, p_j \rangle$. Here, $s_{\texttt{out}}$ is the egress gateway of d_k used to reach h_y. For example, host h_5 resides in domain d_1 and is therefore external to c_2. Accordingly, c_2 records the entry $\langle h_5, s_7, 2 \rangle$, indicating that h_5 is reachable via gateway s_7 of d_2 through port 2.

The route-info table maintains the routing path of each flow $f_{x,y}$ within or traversing domain d_k. For a DI flow, c_k records a route that starts from h_x's switch if $\Phi(h_x) = d_k$ or ends at h_y's switch if $\Phi(h_y) = d_k$. Consider c_2 as an example. The entry for flow $f_{9,8}$ is $\langle s_7, s_5, s_6 \rangle$, corresponding to the route $h_9 \rightarrow s_7 \rightarrow s_5 \rightarrow s_6 \rightarrow h_8$.

For flow $f_{10,11}$, c_2 records an entry $\langle s_8 \rangle$. Since h_{11} resides in domain d_3, c_2 transmits a *cross_relay* message to c_3 with the relevant parameters, including $d_{\texttt{pub}} = d_2$, $\Phi(h_x) = d_2$, $\Phi(h_y) = d_3$, $s_{\texttt{in}} =$ null, $s_{\texttt{out}} = s_8$, $h_x = h_{10}$, $h_y = h_{11}$, $b_{10,11}$, and $\zeta^{\text{U}}_{10,11}$. Upon receiving this message, c_3 adds the entry $\langle s_9, s_{11} \rangle$ to its route-info table for $f_{10,11}$.

For an LB flow $f_{x,y}$, c_k records the route from ingress $s_{\texttt{in}}$ to egress $s_{\texttt{out}}$ in d_k, depending on the flow type:

- **Type I:** $s_{\texttt{in}}$ and $s_{\texttt{out}}$ serve as the entry and exit points connecting the preceding and succeeding domains. For example, if d_1 and d_3 are not neighbors, the entry for flow $f_{1,11}$ in c_2's route-info table is $\langle s_7, s_8 \rangle$, where s_7 and s_8 are gateways used to reach d_1 and d_3, respectively.
- **Type II:** $s_{\texttt{in}}$ and $s_{\texttt{out}}$ connect the egress and ingress gateways of the domain that lends links. For instance, suppose c_3 borrows links from d_2 to route $f_{11,12}$, and c_2 lends link (s_8, s_6). Hence, c_2 records the entry $\langle s_8, s_6 \rangle$ for $f_{11,12}$, where s_8 connects d_3's egress gateway s_9, and s_6 connects d_3's ingress gateway s_{10}.

For each flow, the flow-info table records its attributes (e.g., unit profit, bandwidth demand, and cumulative transmission time for flow duration), per-switch delivered packets in d_k, and domain-level PLR. Table 5 shows an example. For $f_{7,10}$, the unit profit, bandwidth demand, and cumulative transmission time are 3, 5 Mbps, and 30 s, respectively. The route is $h_7 \rightarrow s_5 \rightarrow s_7 \rightarrow s_8 \rightarrow h_{10}$. During the previous period, s_5, s_7, and s_8 delivered 3265, 3241, and 2607 packets, respectively, yielding a PLR in d_2 of $(1 - \frac{2607}{3265}) \times 100\% \approx 20.15\%$.

Theorem 1. In an MDS network with η_{D} domains, domain d_k has η_{H} hosts (including external hosts for communication), η_{S} switches, η_{F} flows, and diameter α. Storing ID and numerical fields requires $\mu_{\texttt{ID}}$ and $\mu_{\texttt{num}}$ bits. The memory overhead per table for controller c_k is: domain-info table $2\eta_{\text{D}}(\mu_{\texttt{ID}}+\mu_{\texttt{num}})$ bits; host-info table $\eta_{\text{H}}(2\mu_{\texttt{ID}} + \mu_{\texttt{num}})$ bits; route-info table $(\alpha + 1)\eta_{\text{F}}\mu_{\texttt{ID}}$ bits; and flow-info table $\eta_{\text{F}}(\mu_{\texttt{ID}} + (\eta_{\text{S}} + 4)\mu_{\texttt{num}})$ bits.

Proof: In the domain-info table, each entry requires two ID fields ($d_{\texttt{tar}}$ and $d_{\texttt{neg}}$) and two numeric fields (m_j, q_j). With at most η_{D} entries, the overhead is $2\eta_{\text{D}}(\mu_{\texttt{ID}} + \mu_{\texttt{num}})$ bits.

In the host-info table, each entry contains two ID fields (host and switch) and one numeric field (p_j). With η_{H} entries, the overhead is $\eta_{\text{H}}(2\mu_{\texttt{ID}} + \mu_{\texttt{num}})$ bits.

In the route-info table, each entry includes a flow ID and a sequence of switch IDs. Since intra-domain routes are bounded by the domain diameter, each entry requires at most $(\alpha + 1)\mu_{\texttt{ID}}$ bits, yielding a total overhead of $(\alpha + 1)\eta_{\text{F}}\mu_{\texttt{ID}}$ bits.

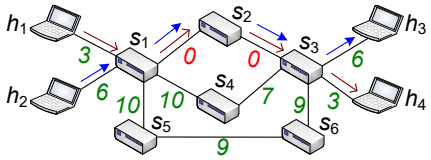


Fig. 3: Example of the flow rerouting process.

In the flow-info table, each entry has a flow ID, four numeric fields (attributes and PLR), and η_{S} per-switch packet counters. With η_{F} entries, the overhead is $\eta_{\text{F}}(\mu_{\texttt{ID}} + (\eta_{\text{S}} + 4)\mu_{\texttt{num}})$ bits. □

4.3 Flow Selection

Controller c_k detects link congestion in domain d_k if a flow $f_{x,y}$ has PLR exceeding δ and multiple flows share congested links in $f_{x,y}$'s route. This can be checked via c_k's flow-info table. The second condition ensures an opportunity to alleviate congestion by coordinating these flows.

Let $\mathcal{F}_\xi$ denote the set of flows on a congested link ξ. If DI flows exist in $\mathcal{F}_\xi$, one is selected for rerouting as discussed in Section 4.4. A DI flow $f_{x,y} \in \mathcal{F}_\xi$ is considered a feasible candidate if it satisfies the following condition:

$$\sum\nolimits_{f_{x',y'} \in \mathcal{F}_\xi} b_{x',y'} - b_{x,y} \leq \theta(\xi). \tag{10}$$

Eq. (10) ensures that rerouting $f_{x,y}$ keeps the total bandwidth of remaining flows in $\mathcal{F}_\xi$ within the link capacity, alleviating congestion. Among all candidates, we select the DI flow with the minimum bandwidth demand to increase the likelihood of finding out an alternative route. For example, suppose that link ξ has 10 Mbps capacity and $\mathcal{F}_\xi$ contains three DI flows, f_1, f_2, and f_3, requiring 6, 5, and 3 Mbps, respectively. Both f_1 and f_2 meet the condition, and f_2 is selected for rerouting.

If no feasible candidate exists, congestion on ξ cannot be resolved by rerouting a single DI flow. In this case, we select the DI flow with the largest profit loss, which is then rerouted along a more favorable path to improve OAP.

If $\mathcal{F}_\xi$ contains only LB flows, no flow is rerouted; instead, the bandwidth of each flow on ξ is constrained, as described in Sections 4.5 and 4.6.

4.4 Flow Rerouting

When controller c_k selects a DI flow $f_{x,y}$ on a congested link ξ, it finds an alternative route within domain d_k satisfying three conditions: 1) the route excludes ξ, detouring the congested link; 2) its hop count increases by at most λ relative to the original route, limiting packet latency; and 3) its *bottleneck bandwidth* (the minimum residual bandwidth along the route) is at least $b_{x,y}$, meeting the flow's demand. If multiple routes satisfy these conditions, the shortest route is chosen.

Consider the example in Fig. 3, where link capacity is set to 10 Mbps and numbers indicate residual bandwidth (some hosts and flows are omitted). Two DI flows, $f_{1,4}$ and $f_{2,3}$, require 7 Mbps and 4 Mbps, respectively, and share links (s_1, s_2) and (s_2, s_3), causing congestion. According to Section 4.3, $f_{2,3}$ is selected for rerouting. With $\lambda = 1$, there are two alternative routes: 1) $h_2 \rightarrow s_1 \rightarrow s_4 \rightarrow s_3 \rightarrow h_3$ (4 hops), and 2) $h_2 \rightarrow s_1 \rightarrow s_5 \rightarrow s_6 \rightarrow s_3 \rightarrow h_3$ (5 hops). We thus replace $f_{2,3}$'s original route with the first alternative.

If c_k cannot find an alternative route for $f_{x,y}$ within d_k, it requests assistance from neighboring domains by sending an ask_help message. Each neighbor c_l responds with a can_help message if its domain d_l has lendable links; otherwise, the message is ignored.

Controller c_k then borrows links from the first domain to reply. If no reply is received before the timeout, no domains can lend links, and c_k must limit the bandwidth of flows on congested links.

Consider $f_{1,4}$ in Fig. 1, with a bandwidth demand of 4 Mbps and a unit profit of 5. If link (s_2, s_4) experiences congestion, no alternative route exists in d_1. The closest gateways to h_1 and h_4 are s_1 and s_4, respectively. Consequently, c_1 publishes a message: "type=ask_help, $d_{\texttt{pub}} = d_1$, $s_{\texttt{in}} = s_4$, $s_{\texttt{out}} = s_1$, $h_x = h_1$, $h_y = h_4$, $b_{x,y} = 4$, $\zeta^{\mathrm{U}}_{x,y} = 5$, $\sigma = 0.3$, $t_{\texttt{bor}} = 30$." Here, the payment ratio is set to 0.3, and the expected duration of link borrowing is 30 s.

Upon receiving the message, controller c_2 checks whether d_2 has lendable links. Two possible routes exist: 1) $s_7 \rightarrow s_8$ and 2) $s_7 \rightarrow s_5 \rightarrow s_6 \rightarrow s_8$. If both can accommodate $f_{1,4}$ (bottleneck bandwidth ≥ 4 Mbps), c_2 selects the shorter route and replies to c_1 with a message: "type=can_help, $d_{\texttt{pub}} = d_2$, $s_{\texttt{in}} = s_7$, $s_{\texttt{out}} = s_8$, $h_x = h_1$, $h_y = h_4$." As a result, $f_{1,4}$'s new route becomes $h_1 \rightarrow s_1 \rightarrow s_7 \rightarrow s_8 \rightarrow s_4 \rightarrow h_4$.

The borrowing duration is specified by $t_{\texttt{bor}}$ in the ask_help message issued by controller c_k. When $t_{\texttt{bor}}$ expires while flow $f_{x,y}$ persists, two cases arise: if domain d_k has an alternative path, c_k switches to it without further borrowing; otherwise, c_k issues a new ask_help message to neighboring domains.

After the transmission of $f_{x,y}$ is completed, the compensation paid by c_k to c_l for link borrowing is given by

$$\zeta_{x,y} \cdot \Theta(f_{x,y}) \cdot \sigma \cdot \left(t^l_{x,y}/t_{x,y}\right), \tag{11}$$

where $t^l_{x,y}$ denotes the total borrowing duration of $f_{x,y}$ in d_l, possibly accumulated over multiple borrowing instances. Here, $\zeta_{x,y} \cdot \Theta(f_{x,y})$ denotes the effective profit obtained by c_k from $f_{x,y}$, and the compensation is proportional to the fraction of borrowing time over the total flow duration.

Substituting Eq. (1) into Eq. (11), the expression reduces to

$$\sigma \zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y} \cdot t^l_{x,y} \cdot \Theta(f_{x,y}). \tag{12}$$

Continuing the above example, suppose $f_{1,4}$ generates 1000 packets, among which 200 are lost. The flow duration of $f_{1,4}$ is 100 s, and it uses links in domain d_2 for a total of 50 s. Option 1 is adopted for $\Theta(f_{x,y})$ (see Eq. (8)). After the transmission of $f_{1,4}$ is completed, c_1 pays c_2 a compensation of $0.3 \times 5 \times 4 \times 50 \times \left(1 - \frac{200}{1000}\right) = 240$. The net profit obtained by c_1 is therefore $5 \times 4 \times 100 \times \left(1 - \frac{200}{1000}\right) - 240 = 1360$.

Regarding the configuration of σ, as the flow duration $t_{x,y}$ is typically known only after $f_{x,y}$ completes, we focus on a single borrowing instance. According to Eq. (12) (where $t^l_{x,y}$ is replaced by $t_{\texttt{bor}}$), the compensation paid by c_k to c_l is upper-bounded by a σ fraction of the effective profit of $f_{x,y}$.

Although σ can be configured based on application requirements, we provide the following practical guidelines:

- $\sigma \leq 0.5$ to avoid excessive compensation.
- σ should not be too small (e.g., $\sigma < 0.1$), as it may lead to insufficient bandwidth allocation when congestion occurs on borrowed links (see Section 4.5).
- For flows with higher unit profit $\zeta^{\mathrm{U}}_{x,y}$, larger bandwidth demand $b_{x,y}$, or longer elapsed time (i.e., larger observed $t_{x,y}$), the overall profit increases (see Eq. (1)). In such cases, a larger σ value helps keep relative compensation low while enabling more bandwidth under congestion.

4.5 Bandwidth Control: NB Module

The NB module restricts the bandwidth usage of type-II LB flows on congested links. Suppose that controller c_k borrows links from domain d_l for flow $f_{x,y}$, and $f_{x,y}$ is a type-II LB flow from c_l's viewpoint. If some borrowed links in d_l become congested (e.g., due to newly generated flows utilizing these links), c_l selects the most congested link among them as the representative, denoted by ξ. This is because ξ determines the bottleneck bandwidth of $f_{x,y}$ over the borrowed links in d_l. Then, c_l calculates the amount of ξ's bandwidth allocated to $f_{x,y}$, which is also the amount of bandwidth that $f_{x,y}$ obtains in d_l (i.e., $b^l_{x,y}$).

The calculation is based on an NB game. Consider a set $\mathcal{Z}$ of players competing for a resource. For each player $z_i \in \mathcal{Z}$, let u_i denote its utility. If players cannot cooperate, z_i obtains a minimal payoff g_i (called a *disagreement point*). Otherwise, the feasible utility region is $\mathcal{U} = \{u_i \,|\, u_i \geq g_i, \forall z_i \in \mathcal{Z}\}$. The NB solution is then given by

$$\mathcal{U}^* = \arg\max_{\mathcal{U}} \prod_{i=1}^{|\mathcal{Z}|} (u_i - g_i), \quad \text{s.t.} \quad u_i \geq g_i. \tag{13}$$

This solution maximizes the product of players' utility gains over their disagreement points.

We consider two players: $z_1 = c_k$ (borrower) and $z_2 = c_l$ (lender). Let congestion occur at time $t_{\texttt{elap}}$, where $t_{\texttt{elap}} < t_{\texttt{bor}}$. As congestion may terminate before the borrowing period expires, the congestion duration is upper-bounded by $t_{\texttt{con}} = t_{\texttt{bor}} - t_{\texttt{elap}}$.

For player 1, its utility captures the portion of $f_{x,y}$'s profit retained by c_k. With Eq. (12), it is given by $u_1 = (1-\sigma)\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y} \cdot t_{\texttt{con}} \cdot \Theta(f_{x,y})$. We consider the general case for $\Theta(f_{x,y})$ (see Eq. (8)). As $f_{x,y}$ is still being transmitted, its exact PLR cannot be determined. Given that $b^l_{x,y} < b_{x,y}$, the proportion of successfully delivered packets (i.e., $1 - n^{\texttt{LOS}}_{x,y}/n^{\texttt{GEN}}_{x,y}$) can be approximated by $b^l_{x,y}/b_{x,y}$. Accordingly, u_1 reduces to

$$u_1 = (1-\sigma)\zeta^{\mathrm{U}}_{x,y} \cdot b^l_{x,y} \cdot t_{\texttt{con}}. \tag{14}$$

Without cooperation, c_k cannot borrow links from domain d_l to reroute $f_{x,y}$ and thus obtains no benefit from c_l. Hence, the disagreement point of player 1 is $g_1 = 0$.

Let $\mathcal{F}_\xi$ be the set of flows on link ξ. For player 2 (i.e., c_l), when it lends ξ to c_k, it may lose a portion of profit that would have been obtained from the flows in $\mathcal{F}_\xi \setminus \{f_{x,y}\}$ (denoted by $\mathcal{F}'_\xi$). Hence, player 2 treats these flows collectively when negotiating with player 1.

Specifically, the total bandwidth demand of these flows is $\tilde{b_l} = \sum_{f_{x',y'} \in \mathcal{F}'_\xi} b_{x',y'}$, and their maximum achievable profit is $\tilde{\zeta_l} = \sum_{f_{x',y'} \in \mathcal{F}'_\xi} \zeta^{\mathrm{U}}_{x',y'} \cdot b_{x',y'} \cdot t_{\texttt{con}}$. Given link capacity $\theta(\xi)$, the utility of player 2 is formulated as follows:

$$u_2 = \tilde{\zeta_l} \cdot \frac{\min\left\{\theta(\xi) - b^l_{x,y}, \tilde{b_l}\right\}}{\tilde{b_l}} + \sigma\zeta^{\mathrm{U}}_{x,y} \cdot b^l_{x,y} \cdot t_{\texttt{con}}. \tag{15}$$

The first term is the actual profit from flows in $\mathcal{F}'_\xi$, while the second term is the compensation paid by c_k for borrowing ξ. This is based on a proportional bandwidth allocation model, where the realized profit is proportional to the fraction of bandwidth supported by the residual link capacity.

If player 2 does not cooperate, it can only obtain profit from flows in $\mathcal{F}'_\xi$, with $b^l_{x,y} = 0$; hence, its disagreement point is

$$g_2 = \tilde{\zeta_l} \cdot (\min\{\theta(\xi), \tilde{b_l}\}/\tilde{b_l}). \tag{16}$$

Substituting u_1, g_1, u_2, and g_2 into Eq. (13) yields the NB solution, with $b^l_{x,y}$ as the only decision variable, which can be numerically obtained via `scipy.optimize` in Python [34].

Theorem 2. Let controller c_l lend out η_{L} links in d_l to reroute flow $f_{x,y}$. The NB module computes $b^l_{x,y}$ in $O(\eta_{\mathrm{L}} + \eta_\xi)$ time, where η_ξ is the number of flows on the most congested link.

Algorithm 1: The BA module

1 Compute $\sum_{f_{x',y'} \in \mathcal{F}'_\xi} \zeta^{\mathrm{U}}_{x',y'} \cdot b_{x',y'}$; $\theta_{\mathtt{A}'}(\xi) \leftarrow \theta_{\mathtt{A}}(\xi)$;
2 **foreach** $f_{x,y} \in \mathcal{F}'_\xi$ **do**
3 Allocate bandwidth of $b^{\mathtt{A}}_{x,y}$ to $f_{x,y}$ via Eq. (17);
4 $\theta_{\mathtt{A}'}(\xi) \leftarrow \theta_{\mathtt{A}'}(\xi) - b^{\mathtt{A}}_{x,y}$;
5 **if** $b^{\mathtt{A}}_{x,y} = b_{x,y}$ **then**
6 $\mathcal{F}'_\xi \leftarrow \mathcal{F}'_\xi \setminus \{f_{x,y}\}$;
7 **if** $\theta_{\mathtt{A}'}(\xi) = 0$ **then**
8 Terminate;
9 Sort flows in $\mathcal{F}'_\xi$ decreasingly by their $\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y}$ values;
10 **foreach** $f_{x,y} \in \mathcal{F}'_\xi$ **do**
11 Allocate additional bandwidth of $\min\{b_{x,y} - b^{\mathtt{A}}_{x,y}, \theta_{\mathtt{A}'}(\xi)\}$ to $f_{x,y}$;
12 $\theta_{\mathtt{A}'}(\xi) \leftarrow \theta_{\mathtt{A}'}(\xi) - \min\{b_{x,y} - b^{\mathtt{A}}_{x,y}, \theta_{\mathtt{A}'}(\xi)\}$;
13 **if** $\theta_{\mathtt{A}'}(\xi) = 0$ **then**
14 Terminate;

Proof: The NB module first identifies the most congested link ξ in $O(\eta_{\mathrm{L}})$ time. It computes u_1 and g_1 in $O(1)$ time, and u_2 and g_2 in $O(\eta_\xi)$ time. The NB problem then reduces to a one-dimensional optimization over $b^l_{x,y}$, solvable in constant time. Thus, the overall time complexity is $O(\eta_{\mathrm{L}} + \eta_\xi)$. □

4.6 Bandwidth Control: BA Module

Complementing the NB module, this module allocates the available bandwidth of a congested link ξ to the flows in $\mathcal{F}_\xi$, excluding type-II LB flows. The resulting set is denoted by $\mathcal{F}'_\xi$, where $\mathcal{F}'_\xi \subseteq \mathcal{F}_\xi$. The available bandwidth $\theta_{\mathtt{A}}(\xi)$ is defined as the capacity of ξ minus the portion allocated to type-II LB flows. Thus, without type-II LB flows on ξ, $\theta_{\mathtt{A}}(\xi) = \theta(\xi)$.

The BA module allocates bandwidth proportionally to the maximum achievable profit of each flow $f_{x,y} \in \mathcal{F}'_\xi$ over the congestion duration $t_{\mathtt{con}}$. By Eq. (1), the profit is $\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y} \cdot t_{\mathtt{con}}$. Since $t_{\mathtt{con}}$ is identical for all flows, the allocation is proportional to $\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y}$. Thus, the bandwidth allocated to $f_{x,y}$ is

$$b^{\mathtt{A}}_{x,y} = \min\left\{ \frac{\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y}}{\sum_{f_{x',y'} \in \mathcal{F}'_\xi} \zeta^{\mathrm{U}}_{x',y'} \cdot b_{x',y'}} \cdot \theta_{\mathtt{A}}(\xi), b_{x,y} \right\}. \quad (17)$$

The formulation further incorporates the bandwidth constraint specified in Eq. (7).

Algorithm 1 presents the pseudocode. Line 1 facilitates the computation of Eq. (17) and set $\theta_{\mathtt{A}'}(\xi) = \theta_{\mathtt{A}}(\xi)$. The for-loop in lines 2–6 allocates bandwidth to each flow $f_{x,y}$ in $\mathcal{F}'_\xi$ and reduces the available bandwidth accordingly. If $f_{x,y}$ meets its demand, it is removed from $\mathcal{F}'_\xi$, as shown in lines 5–6.

If no bandwidth remains, the BA module terminates (lines 7–8). Otherwise, the remaining bandwidth is allocated to flows in $\mathcal{F}'_\xi$ in descending order of $\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y}$ to further satisfy their demands, repeating until all bandwidth is fully allocated (lines 10–14).

Consider two flows, f_1 and f_2, with unit profits of 5 and 4 and demands of 5 Mbps and 6 Mbps, respectively. Given a link capacity of 10 Mbps, proportional allocation assigns f_1 $\frac{5\times5}{5\times5+4\times6} \times 10 \approx 5.1$ Mbps, exceeding its demand; thus, it is capped at 5 Mbps. The remaining bandwidth is then allocated to f_2 (lines 9–14), resulting in a total of 5 Mbps.

The above design provides three advantages. First, Eq. (17) eliminates the unknown term $t_{\mathtt{con}}$. Second, flows with larger $\zeta^{\mathrm{U}}_{x,y} \cdot b_{x,y}$, indicating higher profit potential, are allocated more bandwidth, thereby improving OAP. Third, since $\zeta^{\mathrm{U}}_{x,y} > 0$ and $b_{x,y} > 0$, no flow in $\mathcal{F}'_\xi$ is starved.

TABLE 6: Congestion mitigation mechanisms applied to different flows.

category	rerouting	NB module	BA module
DI flows	✓		✓
type-I LB flows			✓
type-II LB flows		✓	

Theorem 3. Given $\eta_{\xi'}$ flows in $\mathcal{F}'_\xi$, the time complexity of the BA module is $O(\eta_{\xi'} \log_2 \eta_{\xi'})$ in the worst case.

Proof: In Algorithm 1, line 1 requires $O(\eta_{\xi'})$ time. The for-loop in lines 2–6 takes $O(\eta_{\xi'})$ time. Sorting the remaining flows in $\mathcal{F}'_\xi$ in line 9 incurs at most $O(\eta_{\xi'} \log_2 \eta_{\xi'})$ time, while the for-loop in lines 10–14 takes $O(\eta_{\xi'})$ time. Thus, the overall time complexity is $O(\eta_{\xi'} \log_2 \eta_{\xi'})$. □

4.7 Design Rationale

In CRM-PF, flows are classified into three categories: 1) DI flows, 2) type-I LB flows, and 3) type-II LB flows. Under congestion in domain d_k, CRM-PF handles them according to the following guidelines:

- DI flows originate from or terminate at d_k; consequently, controller c_k has the highest flexibility in managing them.
- Type-I LB flows must traverse d_k, so c_k is obligated to forward them and treats them equivalently to DI flows when competing for bandwidth on congested links.
- Type-II LB flows are opportunistic; their bandwidth allocation on congested links depends on the payment that they offer as compensation.

Different congestion mitigation mechanisms are applied to flow categories, as summarized in Table 6. While rerouting alleviates congestion by shifting traffic to less loaded paths and enabling path borrowing when internal links are saturated, it incurs non-negligible overhead due to possible rerouting or bandwidth adjustments of other flows. Hence, under the first guideline, CRM-PF restricts rerouting to DI flows only.

Despite the second guideline treating type-I LB flows as equally important as DI flows, we exclude them from rerouting to avoid excessive path extension caused by link borrowing. Consider a flow $f_{x,y}$ traversing K domains ($K \geq 3$): it is a DI flow in $\Phi(h_x)$ and $\Phi(h_y)$, and a type-I LB flow in the remaining domains. If rerouting were allowed for type-I LB flows, the worst case is that all K domains perform link borrowing. This significantly lengthens the end-to-end path due to cumulative path deviation and leads to excessive network-wide resource consumption, as multiple domains must allocate additional paths for $f_{x,y}$.

To handle a selected DI flow $f_{x,y}$, controller c_k first checks whether an alternative intra-domain route exists within d_k for rerouting. Only if no such route is available does it borrow links from a neighboring domain. This design discourages controllers from opportunistically using inter-domain resources to preserve intra-domain bandwidth.

When c_k borrows links from domain d_l for rerouting $f_{x,y}$, the flow is treated by c_l as a type-II LB flow. Accordingly, c_l does not further borrow links for $f_{x,y}$, thereby preventing cascading link borrowing that complicates route management and unnecessarily increases the path length.

The bandwidth-control mechanism applies to all flow categories. Under the second guideline, CRM-PF adopts the BA module for DI and type-I LB flows on congested links. Under the third guideline, it employs the NB module for type-II LB flows, formulated as an NB game incorporating the payment ratio offered by c_k for $f_{x,y}$.

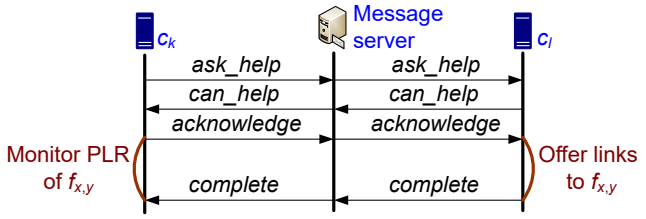


Fig. 4: Message sequence for link borrowing between controllers c_k and c_l.

5 Challenges and Improvements

5.1 Payment Default

When a controller borrows links from another domain to reroute a type-II LB flow, it must pay a portion of the resulting profit to the lending controller after transmission. This design relies on *honest profit sharing* (i.e., no stakeholder defaults), which is impractical. To address this issue, a trusted third party is required to assist with profit payment. Since link borrowing is realized via inter-controller messages relayed by a message server (see Fig. 1), the server is well suited for this role.

Fig. 4 illustrates the message sequence in which c_k borrows links from c_l via the server to reroute $f_{x,y}$ and enable profit payment. First, c_k sends an ask_help message with a specified payment ratio, recorded by the server. If c_l agrees, it replies with a can_help message. After c_k selects c_l, it then sends an acknowledge message, and the server confirms the borrowing relationship. At the end of the borrowing period, c_l sends a complete message to finish the transaction.

To prevent payment default, the server can act as a banking entity that manages each controller's effective profit and executes payments, thereby eliminating the need for the honest profit-sharing assumption.

5.2 Stability

If multiple controllers attempt to borrow links from each other simultaneously, system stability may be affected. To handle this issue, each ask_help message is assigned a timestamp, and link borrowing is determined by timestamp ordering. For example, c_k sends an ask_help message to c_l and then receives one from c_l. Since c_k issued its request earlier, c_l provides link borrowing to c_k, while c_k does not grant borrowing to c_l (i.e., no can_help message is sent). Since inter-controller messages are relayed via the message server, each ask_help message has a unique timestamp, thereby ensuring system stability.

5.3 Robustness

When controller c_k borrows links in domain d_l to serve flow $f_{x,y}$, it specifies the borrowing duration $t_{\texttt{bor}}$ in the ask_help message sent to c_l. If $t_{\texttt{bor}}$ expires while $f_{x,y}$ still requires the borrowed path, c_k must reissue the request (see Section 4.4). This mechanism provides two benefits: 1) it enables c_k to seek alternative domains for rerouting in case of congestion in d_l, and 2) it bounds the service disruption of $f_{x,y}$ by at most $t_{\texttt{bor}}$ in the event of failures in d_l.

Failures in d_l fall into two categories. First, if some borrowed links fail, c_l attempts to find an alternative path within d_l; otherwise, it notifies c_k to reroute $f_{x,y}$ back to d_k. Second, if c_l fails, c_k can employ the following detection schemes:

- **Active detection:** c_k periodically probes c_l to verify its liveness, at the cost of additional message overhead.
- **Passive detection:** c_k keeps monitoring $f_{x,y}$'s PLR; if it exceeds a threshold, c_k probes c_l to confirm its status.

TABLE 7: Software tools and their versions used in the simulation.

component	software tool	version
operating system	Ubuntu	20.04 LTS
network environment	Mininet	2.3.0
SDN controller	Ryu	3.24
switch (data plane)	Open vSwitch	3.5.0
southbound protocol	OpenFlow	1.3
traffic generation	iPerf	2.0.5

Upon detecting a failure of c_l, c_k immediately redirects $f_{x,y}$ back to its domain d_k, thereby reducing disruption time and improving robustness. Since d_l fails to fulfill its obligation, c_k is exempt from paying any profit for this borrowing instance.

5.4 Centralized Bottleneck and Single Point of Failure

In the CRM-PF framework, a centralized message server is responsible for relaying messages among controllers and maintaining the profit accounting mechanism used for borrowing-link transactions. While this design simplifies inter-domain coordination, the message server may become a communication bottleneck as the number of domains and service requests increases. Moreover, the failure of the message server could disrupt both inter-domain communications and profit settlement, thereby affecting the operation of the entire framework.

To address these issues, multiple message servers may be deployed in a distributed and redundant manner. In such an architecture, message forwarding and profit-account management could be replicated across multiple servers, eliminating the dependency on a single server instance. Load-balancing mechanisms can further distribute communication requests among message servers to improve scalability. In addition, consensus-based replication mechanisms, such as Raft [35], can be employed to maintain consistent profit-account information across replicated message servers. Consequently, the failure of an individual message server would not interrupt inter-domain communications or transaction processing, thereby enhancing both system availability and fault tolerance.

6 Performance Evaluation

To evaluate system performance, we developed a simulation environment using the software tools listed in Table 7. More specifically, the simulation was conducted on a virtual machine executing Ubuntu, equipped with four CPU cores and 32 GB of memory. The network environment was emulated via Mininet [36], a widely used framework that enables the virtualization of network nodes (e.g., hosts and switches) within the Linux kernel. The SDN control plane was implemented based on the Ryu framework [37], whereas the data plane was implemented using Open vSwitch [38].

Traffic flows were generated using iPerf [39], with each flow configured as a UDP flow with a packet size of 1470 bytes. Although iPerf supports both TCP and UDP traffic, and the proposed CRM-PF framework is not restricted to UDP, TCP inherently incorporates congestion control and retransmission mechanisms that may obscure the evaluation of congestion mitigation. Specifically, it becomes difficult to isolate whether performance improvements are attributable to TCP's built-in mechanisms or to the proposed framework. Hence, only UDP flows were used to ensure a clear and unbiased evaluation.

Fig. 5 illustrates the MDS topology used in the simulation, which is derived from Fig. 1. The original three domains have been expanded to five, with switches increased from 12 to 18 and hosts from 14 to 20 to evaluate the scalability of the proposed framework. Two types of link capacities are used, where some links

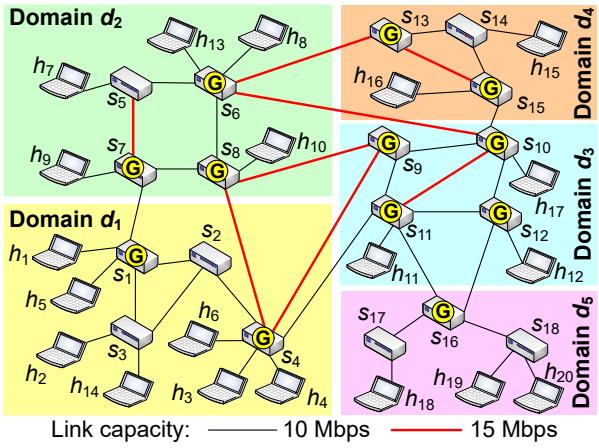


Fig. 5: MDS topology used in the simulation.

are configured with 15 Mbps and others with 10 Mbps, resulting in heterogeneous bandwidths.

Three methods are selected for comparison, including the default OpenFlow-based approach and two MDS-specific flow management methods (discussed in Section 2.3):

- OpenFlow: This baseline approach handles congestion by seeking alternative routes within the local domain only. Links from neighboring domains are not used, i.e., type-II LB flows are not supported.
- *Cooperative flow management (CFM)* [30]: This method supports type-II LB flows by rerouting flows to neighboring domains with spare links. However, if the PLR cannot be reduced due to congestion on the borrowed links, the flow is redirected to its original domain.
- *Multi-domain collaborative route management (MCRM)* [31]: This method considers flow priorities for bandwidth allocation on congested links. Flows are classified into five priority levels based on their unit profits, with type-II LB flows downgraded in the lender's domain.

Next, two experiments are conducted to compare the performance of the proposed CRM-PF framework with the OpenFlow, CFM, and MCRM methods. The first experiment evaluates system performance under different congestion scenarios, while the second examines the impact of varying relationships between bandwidth demands and unit profits of flows.

6.1 Comparison under Varying Congestion Scenarios

In this experiment, multiple flows are generated to induce congestion on links within a domain (specifically, d_1), where no alternative intra-domain routes are available. For each flow, the unit profit is selected from $[1, 10]$, and the bandwidth demand is selected from $[3, 9]$ Mbps. Four congestion scenarios are designed, and Table 8 summarizes their configurations:

- **CS1.** Flows $f_{4,2}$ and $f_{3,1}$ cause congestion in d_1, while d_2 can provide spare links for rerouting.
- **CS2.** Flows $f_{5,6}$, $f_{2,3}$, and $f_{14,3}$ vie for bandwidth in d_1, but no neighboring domain can provide additional links.
- **CS3.** Domain d_2 offers links for flows in d_1; however, some borrowed links subsequently become congested due to the generation of flow $f_{7,13}$ in d_2.
- **CS4.** This scenario extends CS3 by adding two flows, $f_{5,4}$ and $f_{14,3}$, into d_1, further exacerbating congestion.

TABLE 8: Flow configurations under varying congestion scenarios.

Congestion scenario CS1

flow	domains	unit profit	bandwidth	start/end (duration)
$f_{4,2}$	d_1	10	9 Mbps	10/210 (200 s)
$f_{10,12}$	d_2, d_3	2	3 Mbps	20/170 (150 s)
$f_{3,1}$	d_1	8	9 Mbps	35/250 (215 s)
$f_{7,8}$	d_2	3	7 Mbps	50/250 (200 s)
$f_{15,20}$	d_3, d_4, d_5	5	5 Mbps	100/250 (150 s)
$f_{18,19}$	d_5	4	7 Mbps	100/250 (150 s)

Congestion scenario CS2

flow	domains	unit profit	bandwidth	start/end (duration)
$f_{7,8}$	d_2	5	9 Mbps	5/235 (230 s)
$f_{10,9}$	d_2	6	9 Mbps	10/245 (235 s)
$f_{5,6}$	d_1	10	6 Mbps	20/250 (230 s)
$f_{2,3}$	d_1	1	7 Mbps	40/240 (200 s)
$f_{14,3}$	d_1	4	3 Mbps	50/250 (200 s)
$f_{4,12}$	d_1, d_3	2	3 Mbps	70/245 (175 s)
$f_{15,20}$	d_3, d_4, d_5	5	5 Mbps	100/250 (150 s)
$f_{18,19}$	d_5	4	7 Mbps	100/250 (150 s)

Congestion scenario CS3

flow	domains	unit profit	bandwidth	start/end (duration)
$f_{1,6}$	d_1	10	9 Mbps	5/235 (230 s)
$f_{9,10}$	d_2	5	9 Mbps	15/245 (230 s)
$f_{2,3}$	d_1	2	5 Mbps	25/245 (220 s)
$f_{7,13}$	d_2	8	7 Mbps	50/230 (180 s)
$f_{15,20}$	d_3, d_4, d_5	5	5 Mbps	100/250 (150 s)
$f_{18,19}$	d_5	4	7 Mbps	100/250 (150 s)

Congestion scenario CS4

flow	domains	unit profit	bandwidth	start/end (duration)
$f_{1,6}$	d_1	9	8 Mbps	5/235 (230 s)
$f_{9,10}$	d_2	6	9 Mbps	10/240 (230 s)
$f_{2,3}$	d_1	2	6 Mbps	15/235 (220 s)
$f_{7,13}$	d_2	7	7 Mbps	25/250 (225 s)
$f_{5,4}$	d_1	1	3 Mbps	35/250 (215 s)
$f_{14,3}$	d_1	10	5 Mbps	40/250 (210 s)
$f_{15,20}$	d_3, d_4, d_5	5	5 Mbps	100/250 (150 s)
$f_{18,19}$	d_5	4	7 Mbps	100/250 (150 s)

In each scenario, $f_{15,20}$ traverses three domains ($d_4 \Rightarrow d_3 \Rightarrow d_5$) and competes for link bandwidth with $f_{18,19}$ in d_5.

Based on these scenarios, we evaluate each method in terms of network throughput, average PLR, and OAP.

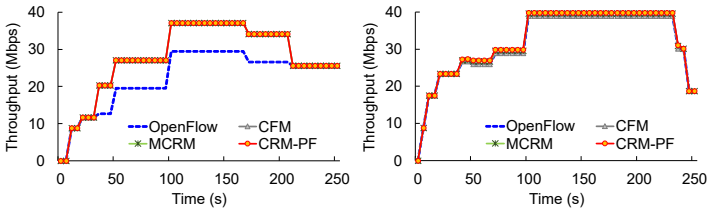
6.1.1 Throughput

Fig. 6(a)–(d) illustrate network throughput over time, which is used to observe how each method reacts to and mitigates congestion. The experiment lasts 250 s with a time interval of 5 s for throughput observation.

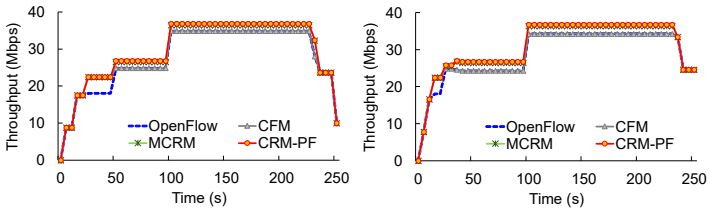
In scenario CS1, $f_{4,2}$ and $f_{3,1}$ are successively introduced into d_1 after the 35th second, leading to congestion at link (s_4, s_2). As shown in Fig. 5, no alternative intra-domain routes exist within d_1 for either $f_{4,2}$ or $f_{3,1}$. Consequently, OpenFlow has no option but to let the congestion persist. In contrast, CFM, MCRM, and CRM-PF reroute one of these flows to d_2, thereby effectively alleviating congestion. This results in a significant throughput gap between OpenFlow and the other three methods after the 35th second.

In scenario CS2, as critical links in d_2 have been occupied by $f_{7,8}$ and $f_{10,9}$, no domain can provide additional links for rerouting flows in d_1. Under this condition, CFM degrades to behavior similar to OpenFlow, where flows compete for bandwidth on congested links. In contrast, MCRM and CRM-PF determine the bandwidth allocated to each flow. However, since the bandwidth of a congested link determines the bottleneck bandwidth of a flow, such bandwidth control is less effective than flow rerouting. As a result, MCRM and CRM-PF achieve only marginal performance improvements over OpenFlow and CFM in this scenario.

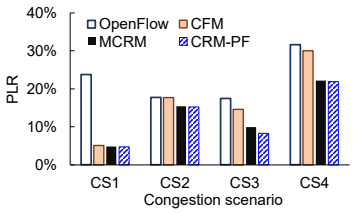
In scenario CS3, from the 25th to the 50th second, d_2 can offer spare links to reroute $f_{1,6}$ or $f_{2,3}$, alleviating congestion in



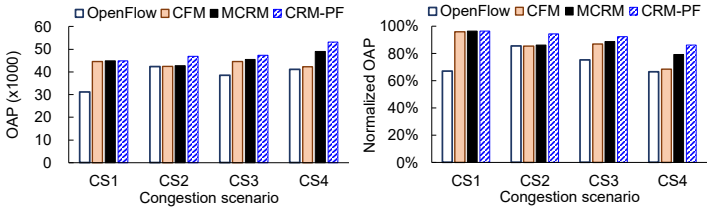
(a) throughput (scenario CS1) (b) throughput (scenario CS2)



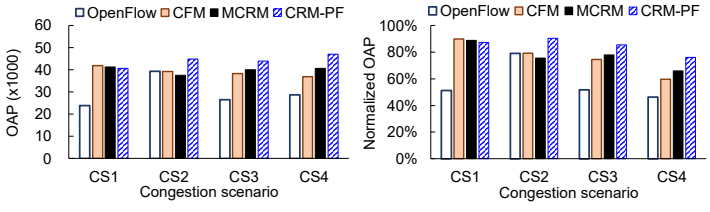
(c) throughput (scenario CS3) (d) throughput (scenario CS4)



(e) average PLR



(f) OAP (Option 1) (g) normalized OAP (Option 1)



(h) OAP (Option 2) (i) normalized OAP (Option 2)

Fig. 6: Performance comparison under varying congestion scenarios.

d_1. Hence, CFM, MCRM, and CRM-PF significantly outperform OpenFlow during this interval. However, after the 50th second, $f_{7,13}$ competes for the borrowed link (s_5, s_6), causing congestion. CFM redirects the affected flow back to d_1 and subsequently behaves similarly to OpenFlow, resulting in comparable performance. In contrast, MCRM and CRM-PF allocate bandwidth among flows on link (s_5, s_6), partially offloading traffic from d_1 and improving overall throughput.

Scenario CS4 is similar to CS3, except that more flows are generated in d_1. Accordingly, the trend in Fig. 6(d) resembles that in Fig. 6(c). Despite the larger number of flows in CS4, the maximum network throughput in both scenarios is comparable, as most flows in d_1 compete for bandwidth on congested links, and the sum of their bottleneck bandwidths approaches link capacity. Consequently, the aggregate bandwidth achieved in d_1 remains similar in CS3 and CS4.

Note that after the 100th second, $f_{15,20}$ and $f_{18,19}$ are added and persist until the 250th second, increasing the overall traffic load and the throughput of all methods in each scenario.

6.1.2 PLR

Fig. 6(e) illustrates the average PLR of all flows, as defined in Eq. (3).

In scenario CS1, OpenFlow keeps $f_{4,2}$ and $f_{3,1}$ in d_1, forcing them into bandwidth contention and yielding an average PLR exceeding 23.7%. In contrast, CFM, MCRM, and CRM-PF reroute one of these flows to d_2, thus alleviating congestion in d_1. Their average PLRs drop below 5.1%, mainly due to residual bandwidth contention between $f_{15,20}$ and $f_{18,19}$.

In scenario CS2, CFM behaves similarly to OpenFlow, leading to flow contention for bandwidth on congested links, resulting in average PLRs exceeding 17.7%. In contrast, both MCRM and CRM-PF explicitly allocate bandwidth per flow, thereby effectively reducing packet loss across flows. Consequently, their average PLRs drop below 15.4%.

In scenario CS3, CFM temporarily reroutes a flow from d_1 to d_2 during the interval 25–50 s, reducing the average PLR from 17.5% under OpenFlow to 14.6%. To handle congestion on shared links, MCRM and CRM-PF apply bandwidth control mechanisms. MCRM allocates bandwidth based on flow priorities, while CRM-PF adopts more sophisticated NB and BA modules. Consequently, CRM-PF reduces the average PLR to 8.3%, compared with 9.9% for MCRM.

Scenario CS4 extends scenario CS3 by adding two flows in d_1, thereby aggravating congestion. As expected, the average PLRs of all methods in this scenario are substantially higher than those observed in the other scenarios. More specifically, the average PLRs of OpenFlow, CFM, MCRM, and CRM-PF are 31.6%, 30.0%, 22.2%, and 21.9%, respectively. It can be observed that, even under severe congestion conditions, CRM-PF achieves the lowest PLR among all methods.

6.1.3 OAP

Fig. 6(f) shows OAP under Option 1. To account for variations in the number of flows and their associated profits, Fig. 6(g) presents the *normalized OAP*, defined as the ratio of OAP to its maximum possible value.

Based on Eq. (8), the effective profit obtained from a flow is inversely proportional to its PLR. As a result, an opposite trend is observed between Fig. 6(e) and Fig. 6(f). Specifically, OpenFlow incurs the highest PLR, followed by CFM, MCRM, and CRM-PF, whereas the highest OAP is observed for CRM-PF, followed by MCRM, CFM, and OpenFlow.

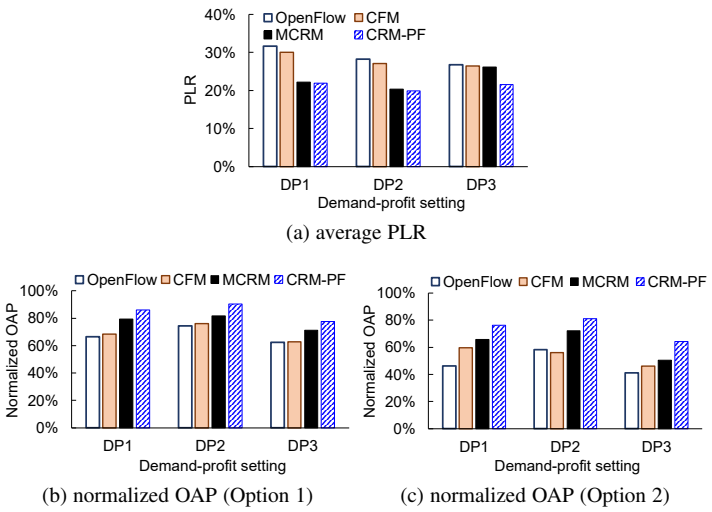
In both scenarios, CS2 and CS4, an interesting observation emerges. Although MCRM and CRM-PF achieve similar average PLRs in Fig. 6(e), a substantial gap is observed in their OAPs, as shown in Fig. 6(f) and Fig. 6(g). This result indicates that the NB and BA modules allow CRM-PF to extract higher profit from flows on congested links than MCRM.

Next, we evaluate the OAP of each method under Option 2. Each flow $f_{x,y}$ is a three-layer SVC flow with one base layer and two enhancement layers. The required layer bandwidths are set to $\tau_0 = 0.5b_{x,y}$, $\tau_1 = 0.3b_{x,y}$, and $\tau_2 = 0.2b_{x,y}$, with weights $w_0 = 0.5$, $w_1 = 0.3$, and $w_2 = 0.2$ in Eq. (9).

Fig. 6(h) presents the OAP, while Fig. 6(i) illustrates the normalized OAP. Compared with Option 1, Option 2 further amplifies the performance gap among different methods, as the step-function-based profit model emphasizes layer completion and makes partial delivery of higher layers more sensitive to packet loss. Hence, methods with better congestion handling and bandwidth allocation mechanisms yield higher OAP.

TABLE 9: Flow configurations under varying demand-profit settings.

flow	DP1	DP2	DP3
$f_{1,6}$	(8 Mbps, 9)	(8 Mbps, 10)	(6 Mbps, 5)
$f_{9,10}$	(9 Mbps, 6)	(8 Mbps, 10)	(9 Mbps, 1)
$f_{2,3}$	(6 Mbps, 2)	(7 Mbps, 8)	(7 Mbps, 4)
$f_{7,13}$	(7 Mbps, 7)	(6 Mbps, 6)	(8 Mbps, 3)
$f_{5,4}$	(3 Mbps, 1)	(4 Mbps, 2)	(5 Mbps, 7)
$f_{14,3}$	(5 Mbps, 10)	(3 Mbps, 1)	(3 Mbps, 10)
$f_{15,20}$	(5 Mbps, 5)	(6 Mbps, 6)	(8 Mbps, 3)
$f_{18,19}$	(7 Mbps, 4)	(5 Mbps, 4)	(4 Mbps, 8)



(a) average PLR

(b) normalized OAP (Option 1)

(c) normalized OAP (Option 2)

Fig. 7: Performance comparison under varying demand-profit settings.

CRM-PF consistently outperforms the other schemes, indicating that although it is not explicitly designed for SVC flows, its mechanism can effectively preserve the delivery of essential SVC layers under congestion, thereby improving OAP.

6.2 Comparison under Varying Demand-Profit Settings

This experiment examines the impact of different demand-profit settings. Scenario CS4 (i.e., the most complex congestion condition) is selected as the representative case. Eight flows are generated, with their domains, start/end times, and durations given in Table 8. Three demand-profit settings are considered, as summarized in Table 9:

- **DP1.** Bandwidth demand and unit profit are independent.
- **DP2.** Unit profit is positively correlated with bandwidth demand; flows with higher demand yield higher profit.
- **DP3.** Unit profit is negatively correlated with bandwidth demand, i.e., the opposite of DP2.

As all settings use the same scenario, their impact on throughput is negligible; thus, we focus on PLR and OAP.

Fig. 7(a) shows the average PLR, following a trend consistent with scenario CS4 in Fig. 6(e): OpenFlow has the highest PLR, followed by CFM, MCRM, and CRM-PF. Under DP3, however, the gap between MCRM and CRM-PF widens due to the negative correlation between unit profit and bandwidth demand. MCRM prioritizes flows solely by unit profit, favoring low-demand flows, whereas CRM-PF accounts for the profit-demand product, alleviating this bias and reducing PLR.

Fig. 7(b) and Fig. 7(c) present the normalized OAP under Options 1 and 2. OAP increases under DP2 due to the positive demand-profit correlation but decreases under DP3 with the negative correlation. CRM-PF consistently attains the highest OAP across all settings, demonstrating its effectiveness.

7 Conclusion and Future Work

An MDS network comprises multiple domains, each managed by an independent controller, avoiding centralized bottlenecks while enabling domain autonomy. To address route management challenges, this paper proposes CRM-PF, which classifies flows into DI, type-I LB, and type-II LB categories and applies tailored congestion mitigation mechanisms. Some DI flows are rerouted to alleviate congestion, while the NB and BA modules regulate bandwidth allocation and enhance profit on congested links without causing flow starvation. Simulation results demonstrate that CRM-PF improves throughput, reduces average PLR, and increases OAP compared to OpenFlow, CFM, and MCRM under varying congestion scenarios and demand-profit settings.

While the CRM-PF framework demonstrates promising performance, several directions remain for future investigation. First, CRM-PF can be extended to multi-administrative MDS environments. Unlike the unified unit-profit model assumed in this paper, independently managed domains may assign different profit values to the same traffic class. Future work will investigate congestion mitigation mechanisms that account for heterogeneous profit models in cross-domain rerouting and link borrowing decisions.

Second, future work will investigate the performance of CRM-PF under TCP traffic. Specifically, the interactions between the BA module in CRM-PF and TCP congestion control warrant further investigation. While the BA module allocates bandwidth on congested links from a network-layer perspective, TCP dynamically adjusts transmission rates through end-to-end congestion control at the transport layer. Evaluating their combined effects may provide valuable insights into cross-layer congestion management and bandwidth allocation in MDS networks.

Third, future work will also validate CRM-PF in a physical hardware testbed. Such an implementation would enable the evaluation of CRM-PF under real-world network conditions and provide further insights into its practicality, scalability, and deployment feasibility in operational MDS environments.

References

[1] L.L. Peterson and B.S. Davie, *Computer Networks: A Systems Approach*. Burlington, Massachusetts: Morgan Kaufmann, 2021.

[2] S. Troia, L. Borgianni, G. Sguotti, S. Giordano, and G. Maier, "A comprehensive survey on software-defined wide area network," *IEEE Comm. Surveys & Tutorials*, vol. 28, pp. 2805–2845, 2026.

[3] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Comm. Surveys & Tutorials*, vol. 20, no. 1, pp. 333–354, 2018.

[4] B.R. Dawadi, D.B. Rawat, S.R. Joshi, and P. Manzoni, "Legacy network integration with SDN-IP implementation towards a multi-domain SoDIP6 network environment," *Electronics*, vol. 9, no. 9, pp. 1–22, 2020.

[5] W.K. Lai, Y.C. Wang, Y.C. Chen, and Z.T. Tsai, "TSSM: Time-sharing switch migration to balance loads of distributed SDN controllers," *IEEE Trans. Network and Service Management*, vol. 19, no. 2, pp. 1585–1597, 2022.

[6] A. Kumari, A. Roy, and A.S. Sairam, "Optimizing SDN controller load balancing using online reinforcement learning," *IEEE Access*, vol. 12, pp. 131591–131604, 2024.

[7] Q. Liu, Y. Liu, Q. Meng, and T. Yu, "CLB-LP: Controller load balancing based on load prediction using deep learning for software-defined IoT networks," *IEEE Trans. Network Science and Engineering*, vol. 12, no. 1, pp. 173–185, 2025.

[8] U. Prajapati, B.C. Chatterjee, and A. Banerjee, "DSFSM: Delay-sensitive fractional switch migration for multi-controller software-defined networks," *Proc. Int'l Conf. Computing, Networking and Comm.*, 2025, pp. 694–698.

[9] A.A. Seyedkolaei, S.A.H. Seno, A. Moradi, and R. Budiarto, "Cost-effective survivable controller placement in software-defined networks," *IEEE Access*, vol. 9, pp. 129130–129140, 2021.

[10] S. Noda, T. Sato, and E. Oki, "Fault-tolerant controller placement model considering load-dependent sojourn time in software-defined network," *IEEE Trans. Network and Service Management*, vol. 20, no. 4, pp. 4887–4908, 2023.

[11] I. Maity, R. Dhiman, and S. Misra, "EnPlace: Energy-aware network partitioning for controller placement in SDN," *IEEE Trans. Green Comm. and Networking*, vol. 7, no. 1, pp. 183–193, 2023.
[12] E. Ballasheni, D. Nace, A. Tomaszewski, and A. Zyle, "A probabilistic optimization approach to the equitable controller location problem," *IEEE Trans. Network and Service Management*, vol. 22, no. 2, pp. 1812–1824, 2025.
[13] I.A. Salti and N. Zhang, "An effective, efficient and scalable link discovery (EESLD) framework for hybrid multi-controller SDN networks," *IEEE Access*, vol.11, pp. 140660–140686, 2023.
[14] Y.C. Wang and S.Y. You, "An efficient route management framework for load balance and overhead reduction in SDN-based data center networks," *IEEE Trans. Network and Service Management*, vol. 15, no. 4, pp. 1422–1434, 2018.
[15] Q. Fu, E. Sun, K. Meng, M. Li, and Y. Zhang, "Deep Q-learning for routing schemes in SDN-based data center networks," *IEEE Access*, vol. 8, pp. 103491–103499, 2020.
[16] Z. Jia, Y. Sun, Q. Liu, S. Dai, and C. Liu, "cRetor: An SDN-based routing scheme for data centers with regular topologies," *IEEE Access*, vol. 8, pp. 116866–116880, 2020.
[17] Y. Liu, H. Gu, Z. Zhou, and N. Wang, "RSLB: Robust and scalable load balancing in software-defined data center networks," *IEEE Trans. Network and Service Management*, vol. 19, no. 4, pp. 4706–4720, 2022.
[18] Y.C. Wang and J.T. Liang, "POFM: Profit-oriented flow management in SDN-based data center networks," *Proc. IEEE Int'l Conf. Industry 4.0, Artificial Intelligence, and Comm. Technology*, 2024, pp. 130–135.
[19] M.N. Pathan, M. Muntaha, S. Sharmin, S. Saha, M.A. Uddin, F.N. Nur, and S. Aryal, "Priority based energy and load aware routing algorithms for SDN enabled data center network," *Computer Networks*, vol. 240, pp. 1–12, 2024.
[20] P. Liu, X. Bai, and H. Cheng, "DRL-MR: Multipath routing based on multi-agent deep reinforcement learning for SDN-based data center networks," *Proc. Int'l Joint Conf. Neural Networks*, 2025, pp. 1–7.
[21] Y. Guo, H. Luo, Z. Wang, X. Yin, and J. Wu, "Routing optimization with path cardinality constraints in a hybrid SDN," *Computer Comm.*, vol. 165, pp. 112–121, 2021.
[22] Y.C. Wang and T.J. Hsiao, "URBM: User-rank-based management of flows in data center networks through SDN," *Proc. Int'l Conf. Computer Comm. and the Internet*, 2022, pp. 142–149.
[23] Y. Zhao, X. Wang, Q. He, C. Zhang, and M. Huang, "PLOFR: An online flow route framework for power saving and load balance in SDN," *IEEE Systems J.*, vol. 15, no. 1, pp. 526–537, 2021.
[24] J. Xu, Y. Wang, B. Zhang, and J. Ma, "A graph reinforcement learning based SDN routing path selection for optimizing long-term revenue," *Future Generation Computer Systems*, vol. 150, pp. 412–423, 2024.
[25] J. Wu and Z. Zhu, "Intelligent routing optimization for SDN based on PPO and GNN," *J. Network and Computer Applications*, vol. 242, pp. 1–12, 2025.
[26] F.X.A. Wibowo, M.A. Gregory, K. Ahmed, and K.M. Gomez, "Multi-domain software defined networking: Research status and challenges," *J. Network and Computer Applications*, vol. 87, pp. 32–45, 2017.
[27] T. Moufakir, M.F. Zhani, A. Gherbi, and O. Bouachir, "Collaborative multi-domain routing in SDN environments," *J. Network and Systems Management*, vol. 30, no. 23, pp. 1–13, 2022.
[28] M. Hata, S. Izumi, T. Abe, and T. Suganuma, "A proposal of SDN based mobility management in multiple domain networks," *Proc. Asia-Pacific Network Operations and Management Symp.*, 2016, pp. 1–4.
[29] A.A.Z. Ibrahim, F. Hashim, A. Sali, N.K. Noordin, and S.M.E. Fadul, "A multi-objective routing mechanism for energy management optimization in SDN multi-control architecture," *IEEE Access*, vol. 10, pp. 20312–20327, 2022.
[30] Y.C. Wang and E.J. Chang, "Cooperative flow management in multi-domain SDN-based networks with multiple controllers," *Proc. IEEE Int'l Conf. Smart Communities: Improving Quality of Life Using ICT, IoT and AI*, 2020, pp. 82–86.
[31] Y.C. Wang and L.C. Yen, "Collaborative route management to mitigate congestion in multi-domain networks using SDN," *Proc. IEEE Annual Computing and Comm. Workshop and Conf.*, 2022, pp. 988–994.
[32] A. Thapa, D. Karki, R. Guragain, S.P. Upadhyaya, B.R. Dawadi, and S.R. Joshi, "Experimental evaluation of a multi-domain routing in SDN and its inter-operability with legacy networks," *Proc. IEEE Int'l Conf. Electronics, Computing and Comm. Technologies*, 2021, pp. 1–6.
[33] ZeroMQ. [Online]. Available: https://zeromq.org/
[34] SciPy API. Optimization and root finding (scipy.optimize). [Online]. Available: https://docs.scipy.org/doc/scipy/reference/optimize.html
[35] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," *Proc. USENIX Annual Technical Conf.*, 2014, pp. 305–320.
[36] Mininet. [Online]. Available: https://mininet.org/
[37] Ryu. [Online]. Available: https://ryu-sdn.org/
[38] Open vSwitch. [Online]. Available: https://docs.openvswitch.org/
[39] iPerf. [Online]. Available: https://iperf.fr/