

一种轻量级自适应数据包调度方法

陈天宇, 王昌达

(江苏大学 计算机科学与通信工程学院, 江苏 镇江 212013)

摘要: 数据包调度是保障网络服务质量的核心机制。在 SDN 网络中, 为实现高性能且轻量化的自适应调度, 提出一种基于 K-means 聚类的数据包调度方法 CISO (Classify-In-Sort-Out)。该方法主要包括 3 个步骤: ①在控制平面利用 K-means 算法对数据包优先级分布进行聚类分析, 生成自适应整数阈值; ②在数据平面通过阈值比较实现数据包分类与初始队列映射; ③结合拥塞感知降级与级联入队机制, 优化队列分配与丢包操作。实验结果表明, CISO 在调度逆序数上相较于 RIFO 和 SP-PIFO 分别降低 3.6 倍和 1.2 倍, 短流完成时间仅较理想 PIFO 增加 5%~9%。此外, 其在数据平面仅需进行整数比较操作, 显著优于 PACKS 等复杂方案, 实现了 SDN 网络中数据包调度精度与资源效率的有效平衡。

关键词: 可编程网络; 数据包调度; 网络内处理; 网络流量; 数据包分类; 机器学习

DOI: 10.11907/rjdk.251729

扫描二维码阅读全文:

中图分类号: TP393

文献标识码: A

文章编号: 1672-7800(XXXX)0XX-0001-07



A Lightweight Adaptive Data Packet Scheduling Method

CHEN Tianyu, WANG Changda

(School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China)

Abstract: Packet scheduling is the core mechanism ensuring network service quality. In SDN networks, to achieve high-performance and lightweight adaptive scheduling, a packet scheduling method based on K-means clustering, named CISO (Classify-In-Sort-Out), is proposed. This method mainly includes three steps: ①utilizing the K-means algorithm in the control plane to perform cluster analysis on packet priority distribution and generate adaptive integer thresholds; ②implementing packet classification and initial queue mapping through threshold comparison in the data plane; ③combining congestion-aware degradation and cascaded enqueue mechanisms to optimize queue allocation and packet loss operations. Experimental results show that CISO reduces the number of scheduling inversions by 3.6 times and 1.2 times compared to RIFO and SP-PIFO, respectively, and only increases the completion time of short flows by 5% to 9% compared to ideal PIFO. Furthermore, it only requires integer comparison operations in the data plane, significantly outperforming complex schemes such as PACKS, achieving an effective balance between packet scheduling accuracy and resource efficiency in SDN networks.

Key Words: programmable networks; packet scheduling; in-network processing; network traffic; packet classification; machine learning

0 引言

随着互联网流量的爆炸式增长和网络服务需求的多样化, 现代网络架构正面临着前所未有的挑战。传统网络设备依赖专用硬件和固定调度算法, 如最短剩余处理时间 (Shortest Remaining Processing Time, SRPT)^[1] 和加权公平队列 (Weighted Fair Queuing, WFQ)^[2], 这些算法通常由专用集成电路 (Application-Specific Integrated Circuits, ASIC)

实现^[3]。尽管这些方法在特定场景下表现优异, 但其硬件依赖导致网络缺乏灵活性和可扩展性, 难以适应快速变化的业务需求。在这一背景下, 软件定义网络 (Software-Defined Networking, SDN) 应运而生。SDN 可将控制平面与数据平面分离, 网络管理员以软件方式动态定义和管理网络行为, 从而实现了网络资源的灵活调度与高效利用^[4]。

随着可编程数据平面技术的发展, 特别是以 P4 (Programming Protocol-Independent Packet Processors) 为代表的协议无关编程模型的出现, 网络设备具备了在数据平面直

收稿日期: 2025-10-25

基金项目: 国家自然科学基金项目 (62572219)

作者简介: 陈天宇 (2001-), 男, 江苏大学计算机科学与通信工程学院硕士研究生, 研究方向为网络数据包调度; 王昌达 (1971-), 男, 博士, 江苏大学计算机科学与通信工程学院教授、博士生导师, 研究方向为网络安全。本文通讯作者: 王昌达。

接执行用户自定义逻辑的能力^[5]。这种能力使得网络不仅能够转发数据包,而且能在数据包级别实现复杂的处理逻辑,如负载均衡、拥塞控制、安全策略执行等,极大拓展了网络智能化的边界^[6]。然而,在数据平面进行编程存在一些挑战。P4作为数据平面编程设计的特定语言,仅支持简单的状态操作。复杂的有状态任务需要分布式状态管理或与控制平面的频繁交互,这两者均会给数据包调度引入延迟。此外,三态内容寻址存储器(Ternary Content-Addressable Memory, TCAM)、寄存器等关键硬件资源有限,调度逻辑必须经过优化,以适应交换机可用表大小和流水线深度,否则可能导致数据包丢失或网络性能下降^[7]。

在众多网络功能中,数据包调度(Packet Scheduling)作为服务质量(Quality of Service, QoS)保障的核心机制,其性能直接影响网络的吞吐量、延迟、公平性和资源利用率^[8]。理想的调度算法应根据数据包的优先级、截止时间或流特征动态调整其转发顺序,从而满足不同应用的服务等级协议(Service-Level Agreement, SLA)。PIFO(Push-In First-Out)队列抽象作为一种理想的调度模型,允许数据包根据其优先级被插入到队列中的任意位置,并在出队时严格按优先级顺序处理,由此实现高度灵活和精确的调度策略^[9-10]。然而,PIFO的实现依赖于全局排序机制,这在高速网络环境中带来了极高的硬件复杂性和资源开销,限制了其在实际系统中的应用。因此,当前研究的焦点集中在如何以可承受的硬件复杂度高效地近似PIFO行为。

1 相关工作

根据设计理念与资源开销特征,现有近似PIFO方案可大致分为高精度、高复杂度的多队列方案,资源高效但精度受限的单队列方案,以及缺乏分布感知能力的静态映射方案3类。第一类方案以PACKS(PACKet Scheduler)^[11]和PR-AQM(Packet Rank-aware Active Queue Management)^[12]为代表,其通过维护多个优先级队列和复杂的统计机制来逼近PIFO行为。该类方法的核心优势为利用流量分布的统计特征实现精细调度,但根本局限在于对分布形态的隐含假设,即当实际流量呈现多峰、偏态或动态变化的复杂模式时,这些基于固定统计量的映射策略难以自适应调整,导致队列分配失衡。更重要的是,复杂的量化计算与大规模状态维护对可编程数据平面的简单运算单元和有限寄存器资源构成巨大挑战,在BMv2(Behavioral Model Version 2)等资源受限环境中显著增加了开销^[13]。第二类方案则偏向于简单的逻辑实现和较少的资源消耗。以RIFO(Range-In First-Out)^[14]和AIFO(Admission-In First-Out)^[15]为例,其仅使用单个FIFO(First-In First-Out)队列,并通过基于秩的准入控制来决定是否接纳新到达的数据包。这种设计仅需极少的逻辑与存储资源,但其根本缺陷在于完全牺牲了数据包的重排序能力。由于所有数

据包无论优先级高低均进入同一个队列,调度器无法及时对高优先级流量进行调度,导致拥塞或突发流量场景下的调度性能显著劣于多队列方案^[16]。第三类方案以SP-PIFO(Strict-Priority Push-In First-Out)^[17]为代表,其使用严格优先级队列来近似PIFO行为,通过静态规则将流量预先分配到不同优先级的队列中。然而,SP-PIFO的映射策略缺乏对实时网络拥塞状态的感知,这会导致两个问题:一是高优先级队列可能被大量非最高优先级的流量占据,造成资源浪费;二是低优先级队列可能因无法获得服务而饥饿。其静态映射机制在面对动态变化的流量模式时,性能与理想PIFO存在一定差距。

上述方法的共性问题在于未能有效解决调度精度、资源开销与分布适应性之间的三重权衡问题,这一矛盾在SDN和边缘计算等场景中尤为突出,因为这些环境既需要应对复杂多变的流量模式,又普遍运行在资源受限的商用硬件或软件交换机上^[18-19]。近年来,在线机器学习在网络数据平面的轻量化应用中展现出一定潜力。如何在不引入高开销的前提下实现有效分布感知仍是关键挑战。为在P4交换机等资源受限的数据平面上以极低的开销高效地近似PIFO的调度行为,同时使调度策略能自动适应动态变化的流量分布,本文提出一种基于K-means的轻量级自适应数据包调度方法CISO(Classify-In-Sort-Out),主要贡献点为:①提出轻量级在线聚类评分机制,通过K-means自动发现流量的自然聚类结构,仅需维护聚类中心与样本计数等有限状态,无需预设分布模型或进行复杂统计计算,实现了对复杂流量模式的自适应感知;②设计了密度驱动的智能队列映射策略,基于聚类发现的分布特征动态分配队列资源,使高密度优先级区域获得更多调度机会,同时结合实时拥塞状态进行细粒度调整,引入最高优先级快速通道保障关键流量的及时处理;③采用级联入队与聚类维护协同机制,在避免不必要丢包的同时通过聚类合并、分裂等操作持续优化分布模型,确保调度策略与流量特征的长期匹配;④在Netbench和BMv2上实现了CISO原型,并验证了其在资源消耗、调度性能等方面的优势。

2 CISO方法构建

2.1 总体设计

CISO采用控制平面与数据平面协同的混合架构。在控制平面,其利用K-means聚类算法对数据包优先级分布进行离线学习,生成一组严格递增的整数阈值;在数据平面,其利用该阈值序列为每个数据包执行轻量级调度决策流程。总体架构如图1所示。

2.2 控制平面:K-means聚类与阈值生成

控制平面的核心任务为将连续的rank值域划分为K个区间,并为每个区间分配一个优先级队列。本研究采用K-means聚类算法实现数据驱动的自适应区间划分。该

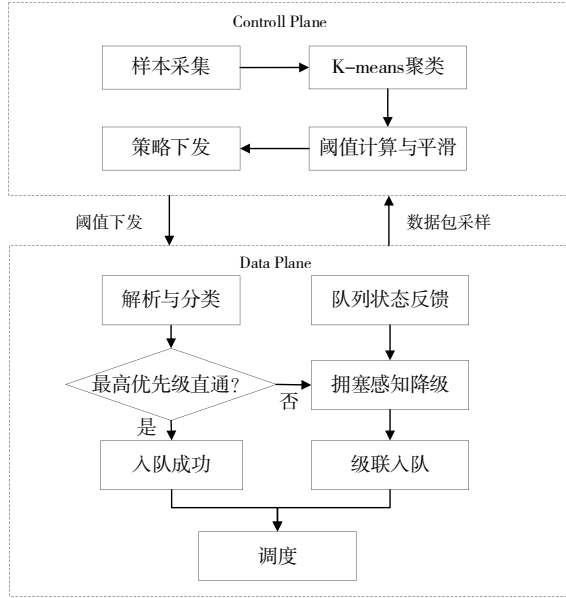


Fig. 1 CISO architecture

图1 CISO架构

算法是一种高效的无监督学习算法,其质心能自动定位到数据包优先级分布中的密度中心。这与传统的静态分位数(如PACKS)或线性划分(如SP-PIFO)有本质区别:静态划分在面对偏态或多峰分布时会失效,导致某些队列极度拥塞而另一些队列空闲;K-means能自适应地在密度高的区域放置更多质心,从而生成更精细的阈值,使得高密度区域的流量能被有效地分散到多个队列中,避免了单点拥塞,实现了更优的负载均衡和调度精度。此外,数据平面的处理能力有限。K-means的迭代、浮点计算和全局状态维护均在控制平面完成,控制平面最终向数据平面下发整数阈值。数据平面的逐包处理流程则被简化为一系列整数比较和寄存器读写操作,这些操作均能在P4流水线中高效完成,从而保证了线速处理能力。

2.2.1 样本采集与聚类

以时间窗口为单位,采集最近通过的 m 个数据包的 $rank$ 值,构成样本集 $X = \{x_1, x_2, \dots, x_m\}$ 。随后,对样本集执行标准的K-means聚类,其目标函数为最小化所有样本到其所属簇质心的平方距离之和。表示为:

$$\min_{\{\mu_k\}_{k=0}^{K-1}, \{z_{ik}\}} \sum_{i=1}^m \sum_{k=0}^{K-1} z_{ik} \|x_i - \mu_k\|^2 \quad (1)$$

约束条件为:

$$\text{s.t. } \sum_{k=0}^{K-1} z_{ik} = 1, z_{ik} \in \{0, 1\} \quad (2)$$

式中: μ_k 为第 k 个簇的质心; z_{ik} 为指示样本 x_i 是否属于簇 k 的二元变量。算法迭代执行分配与更新步骤直至收敛。最终,将得到的 K 个质心按数值升序排列,记为 $\mu_0 < \mu_1 < \dots < \mu_{K-1}$ 。

2.2.2 整数阈值量化与平滑

为了在数据平面进行高效比较,本研究将连续的质心转化为离散的整数阈值。取相邻质心的中点作为分类边界,并将其量化为整数。表示为:

$$\theta_i = \left\lfloor \frac{\mu_{i-1} + \mu_i}{2} \right\rfloor, i = 1, 2, \dots, K-1 \quad (3)$$

由此得到有序的阈值集合 $\Theta = \{\theta_1, \theta_2, \dots, \theta_{K-1}\}$,所有阈值与 $rank$ 字段采用相同位宽。

为避免因流量短时波动导致的阈值频繁抖动,引入指数滑动平均和最小步长约束对阈值进行平滑。设第 t 个周期计算得到的新阈值为 $\hat{\theta}_i^{(t)}$,则实际下发的阈值 $\theta_i^{(t)}$ 按以下公式进行更新:

$$\theta_i^{(t)} \leftarrow \text{round}((1 - \alpha) \cdot \theta_i^{(t-1)} + \alpha \cdot \hat{\theta}_i^{(t)}) \quad (4)$$

式中: $\alpha \in (0, 1)$ 为平滑系数。如果本次更新前后的变化量 $|\theta_i^{(t)} - \theta_i^{(t-1)}|$ 小于预设的最小步长 δ ,则保持阈值不变。该防抖机制能有效降低控制平面与数据平面之间的交互频率与系统开销。最终,控制平面将平滑后的阈值集合 Θ 以及极高优先级阈值 τ_{top} 写入交换机的常量寄存器,完成对数据平面调度策略的更新。设置聚类数目 K 与物理队列数目 N 相等,则类别索引直接对应队列索引。

为使CISO在系统启动或流量模式突变时能快速进入稳定调度状态,本研究引入基于历史流量的K-means预训练机制。在预训练阶段,使用具有代表性的历史流量数据提取数据包优先级 $rank$ 值,构成训练样本集,并执行K-means聚类以获取初始质心,这些质心经过量化与平滑后作为初始阈值下发至数据平面。预训练不仅避免了系统初始阶段的阈值盲区,而且提升了CISO对常见流量模式的适应能力,为后续的在线自适应学习奠定了良好基础。

2.3 数据平面:逐包处理流程

数据平面的处理流程被设计为一条确定性的、无状态的流水线,每个数据包解析得到其整数 $rank$ 值后依次执行以下步骤。

2.3.1 分类与初始队列选择

将数据包的 $rank$ 值 r 与阈值集合 Θ 进行比较,确定其所属的优先级类别。判别公式为:

$$c(r) = \begin{cases} 0, & \text{if } r < \theta_1 \\ 1, & \text{if } \theta_1 \leq r < \theta_2 \\ \vdots & \\ K-1, & \text{if } r \geq \theta_{K-1} \end{cases} \quad (5)$$

得到类别索引 c 后,通过查询映射关系得到初始队列索引 Q_{init} 。

2.3.2 极高优先级直通

为了在网络拥塞时仍能保证最高优先级数据包的及时调度,CISO设置了直通机制。如果数据包满足式(6)所示的条件:

$$(Q_{init} = 0) \wedge (r < \tau_{top}) \quad (6)$$

则其将被直接赋予最终队列索引 $Q_{final} \leftarrow 0$,并跳过后续的拥塞感知降级逻辑。

2.3.3 拥塞感知降级

对于不满足直通条件的数据包,系统将从寄存器中读取其初始队列 Q_{init} 的近似实时长度 $qlen$,并与整数阈值 t_d

进行比较,以决定是否对其进行降级。比较公式为:

$$Q_{final} = \begin{cases} \min(Q_{init} + 1, N - 1), & \text{if } qlen(Q_{init}) > t_d \\ Q_{init}, & \text{otherwise} \end{cases} \quad (7)$$

判断降级所用的拥塞阈值 t_d 由控制平面预计算为整数 $t_d = \lfloor k \cdot B \rfloor$, 其中 k 为降级系数, B 为队列容量。队列长度信息由出口交换机流水线通过专用的工作数据包周期性地读取流量管理器中的实际队列长度, 并再循环至入口交换机流水线的全局寄存器中, 从而为调度决策提供近实时的拥塞状态。

2.3.4 级联入队

最终的入队操作由流量管理器执行, 调度器从 Q_{final} 开始尝试入队, 若队列未满则将数据包入队, 否则继续向更低的队列尝试入队操作。如果所有候选队列均满, 则丢弃该数据包。该机制最大限度地提高了缓冲区利用率, 减少了因单队列瞬时拥塞导致的非必要丢包。

2.4 完整流程

算法 1 和算法 2 综合了上述所有步骤, 给出了数据包调度的完整流程。其中算法 1 负责 CISO 控制平面阈值更新, 伪代码为:

输入: 数据包 rank 值样本 x , 聚类数 k , 平滑系数 α , 最小步长 δ

输出: 更新后的阈值集合 θ_{new}

符号: 聚类中心 μ , 临时阈值 θ_{temp} , 旧阈值 θ_{old} , 平滑后阈值 θ_{smooth}

```

1. function ciso_control_plane_update
2.    $\mu[0..k-1] \leftarrow kmeans\_clustering(x, k)$ 
3.    $sort(\mu)$ 
4.   FOR  $i = 1$  TO  $k-1$  DO
5.      $\theta_{temp} \leftarrow floor((\mu[i-1] + \mu[i]) / 2)$ 
6.      $\theta_{smooth} \leftarrow round((1-\alpha) \times \theta_{old[i]} + \alpha \times \theta_{temp})$ 
7.     IF  $|\theta_{smooth} - \theta_{old[i]}| \geq \delta$  THEN
8.        $\theta_{new[i]} \leftarrow \theta_{smooth}$ 
9.     ELSE
10.       $\theta_{new[i]} \leftarrow \theta_{old[i]}$ 
11.    END IF
12.  END FOR
13.  RETURN  $\theta_{new}$ 
14. END function

```

算法 2 负责 CISO 数据平面逐包处理, 伪代码为:

输入: 数据包 $packet$, 队列数 n , 阈值集合 θ , 拥塞阈值 t_d , 极高优先级门檻 τ_{top}

输出: 数据包被调度到某个队列或丢弃

符号: 数据包 rank 值 r , 类别索引 c , 临时阈值 Q_{init} , 队列长度 $qlen$, 最终队列 Q_{final}

```

1. function ciso_data_plane_processor
2.    $r \leftarrow extractrank(packet)$ 
3.    $c \leftarrow k-1$ 
4.   FOR  $i = 1$  TO  $k-1$  DO
5.     IF  $r < \theta[i]$ 

```

```

6.        $c \leftarrow i-1$ 
7.       BREAK
8.     END IF
9.   END FOR
10.   $Q_{init} \leftarrow c$ 
11.  IF  $(Q_{init} == 0)$  AND  $(r < \tau_{top})$ 
12.     $Q_{final} \leftarrow 0$ 
13.  ELSE
14.     $qlen \leftarrow read\_queue\_len(queue\_len\_regs[Q_{init}])$ 
15.    IF  $qlen > t_d$ 
16.       $Q_{final} \leftarrow \min(Q_{init} + 1, n-1)$ 
17.    ELSE
18.       $Q_{final} \leftarrow Q_{init}$ 
19.    END IF
20.  END IF
21.   $enqueued \leftarrow false$ 
22.  FOR  $Q_{try} = Q_{final}$  TO  $n-1$  DO
23.    IF NOT  $is\_queue\_full(queue[Q_{try}])$  THEN
24.       $enqueue(p, queue[Q_{try}])$ 
25.       $enqueued \leftarrow true$ 
26.      BREAK
27.    END IF
28.  END FOR
29.  IF NOT  $enqueued$  THEN
30.     $drop(packet)$ 
31.  END IF
32. END function

```

2.5 复杂度与资源占用分析

控制平面复杂度主要来源于 K-means 聚类。对于一个包含 m 个样本、迭代 I 轮、寻找 K 个簇的任务, 其时间复杂度为 $O(K \cdot m \cdot I)$, 空间复杂度为 $O(K + m)$ 。阈值平滑与下发的开销为 $O(K)$ 。数据平面的时间复杂度为: 分类阶段需进行 $O(K)$ 次整数比较; 降级判断需 1 次整数比较; 级联入队在最坏情况下进行 $O(N)$ 次队列检查, 但由于队列通常不满, 平均开销为常数。在资源占用方面, 数据平面需要存储 $(K-1)$ 个阈值寄存器、1 个 τ_{top} 寄存器以及 N 个队列长度寄存器, 未引入任何额外的复杂运算单元或大型表项, 对交换机资源友好。

3 实验方法与结果分析

3.1 实验方法

将 CISO 与当前主流高性能调度器 (PIFO^[9]、PACKS^[11]、轻量级调度器 (RIFO^[14]) 及基线方案 (FIFO、SP-PIFO^[17]) 进行比较。评估围绕两个核心目标展开: ① 验证 CISO 在逼近理想 PIFO 调度行为方面的能力; ② 考察 CISO 在混合流量负载下对关键性能指标的优化效果。

在 Netbench 数据包级仿真平台上实现 CISO 及其对比方案。实验采用 leaf-spine 拓扑, 接入与骨干链路带宽分

别为 1 Gbps 与 4 Gbps。为全面评估性能,本研究使用由 pFabric 流量模型生成的混合工作负载,包含大量(约 80%)短流(<100 KB)和少量长流(≥ 1 MB)。数据包的优先级 rank 由其在流中的位置决定,以模拟 SRPT 等策略。CISO、PACKS 与 SP-PIFO 均配置 8 个优先级队列, RIFO 使用等效大小的单队列, PIFO 作为理想基线。CISO 的控制平面聚类周期 T 设置为 1 000 个数据包,平滑系数 $\alpha=0.3$ 。

此外,为验证 CISO 在标准软件交换机环境中的数据平面逻辑,本研究使用 P4 实现 CISO,并在 BMv2 软件交换机上对其进行评估。在 Mininet 中采用简单拓扑,其由 4 个发送方主机、1 个运行 CISO P4 程序的 BMv2 交换机和 1 个接收方主机组成。发送方与交换机、交换机与接收方之间的链路均设置为 1 Gbps。使用流量生成器生成 4 个并发的 UDP 流,每个流配置不同等级值以代表不同优先级,具体为流 1>流 2>流 3>流 4,其中较低的数值等级表示较高的调度优先级。这些流按顺序启动,间隔时间为 10 s。作为比较基线,本研究将 BMv2 交换机配置为标准 FIFO 排队行为,监测接收端每个流随时间变化的吞吐量,以分析带宽分配模式。

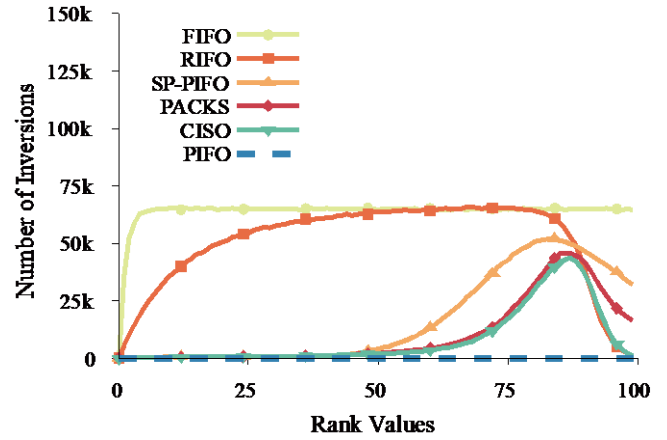
3.2 实验结果与分析

3.2.1 调度逆序与丢包行为分析

调度器的核心任务为保证高优先级数据包优先被服务。首先在均匀优先级分布下统计各方案产生的调度数据包逆序次数。如图 2(a)所示,由于全局排序严格,理想的 PIFO 以准确的顺序调度数据包,逆序数为零。FIFO 和 RIFO 因其单队列结构完全或基本丧失了重排序能力,逆序数最高。SP-PIFO 的静态映射策略在面对动态流量时不够灵活,产生了较多逆序。PACKS 通过复杂的滑动窗口机制实现了较好的近似,逆序数较低。CISO 的表现十分接近于理想的 PIFO,其逆序数相较 RIFO 降低 3.6 倍,相较 SP-PIFO 降低 1.2 倍,验证了控制面 K-means 聚类生成阈值的有效期。与 SP-PIFO 的固定线性区间不同,CISO 通过离线学习产生的阈值能自适应地匹配当前流量的优先级分布,从而在数据面仅通过一系列简单的整数比较便能实现更精细的优先级划分,将 rank 相近的数据包更准确地归入同一队列,最终以极低的复杂度达成高精度的排序效果。

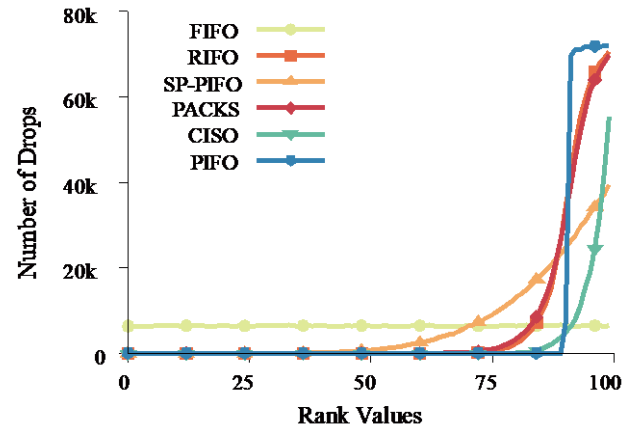
如图 2(b)所示,在丢包行为上,PIFO 无丢包;FIFO 无差别地随机丢包;RIFO 和 PACKS 采用准入控制,主动丢弃大量低优先级数据包;CISO 展现出更温和、更智能的丢包曲线,其丢弃的数据包几乎全部集中在最高优先级(rank>85)的范围内,丢包行为比 PACKS(丢弃秩>79 的包)更接近 PIFO 的最优性能,并显著优于 SP-PIFO(丢弃秩>50 的包)和 FIFO(无差别丢弃)。CISO 的丢包优势源于级联入队机制与精准初始映射的协同作用。当目标队列满时,数据包不会立即被丢弃,而是有机会降级到低优先级队列中暂存;同时,基于 K-means 的精准初始映射从源头减少了

数据包被错误地引向已拥塞队列的概率。两者的结合有效避免了中优先级数据包的意外丢失,实现了智能拥塞缓解。



(a) Scheduling inversions

(a) 调度逆序数



(b) Packet drops

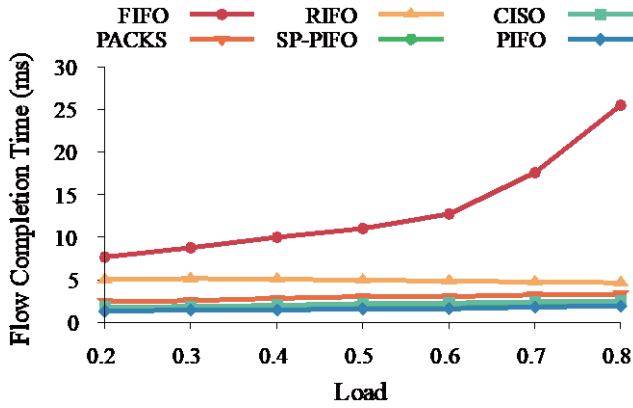
(b) 丢包数

Fig. 2 Scheduling inversion and packet loss behavior analysis

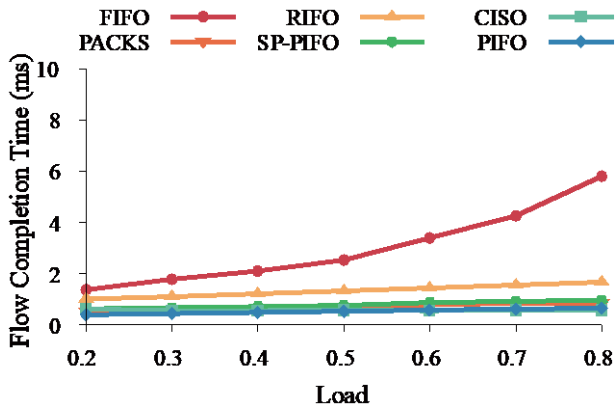
图2 调度逆序与丢包行为分析

3.2.2 不同负载下的流完成时间分析

流完成时间(Flow Completed Time, FCT)是衡量数据中心网络性能的关键指标。本研究分别评估了短流和长流的 FCT。其中,短流对延迟极度敏感。图 3(a)为短流平均 FCT 随负载的变化情况。可以看出,在所有负载下,CISO 的 FCT 均低于 FIFO、RIFO 和 SP-PIFO,其曲线最接近理想的 PIFO。具体而言,在 70% 负载下,CISO 的平均 FCT 相较 RIFO 降低约 65%,相较 SP-PIFO 降低约 15%,仅比 PIFO 提升 7%。图 3(b)为短流尾部 FCT(99th FCT)。可以看出,CISO 的表现同样出色,99th FCT 仅比 PIFO 提升 8%,但相较 RIFO 和 SP-PIFO 分别降低 65% 和 20%。这一卓越表现源于 CISO 动态阈值与极高优先级直通机制的结合。K-means 能精准识别出代表短流的高优先级短流集群,并将其精确映射到最高优先级队列,确保短流即使在高负载情况下也能几乎无阻塞地快速通过。



(a) Average FCT of small flows
(a) 短流平均 FCT



(b) 99th FCT of small flows
(b) 短流尾部 FCT

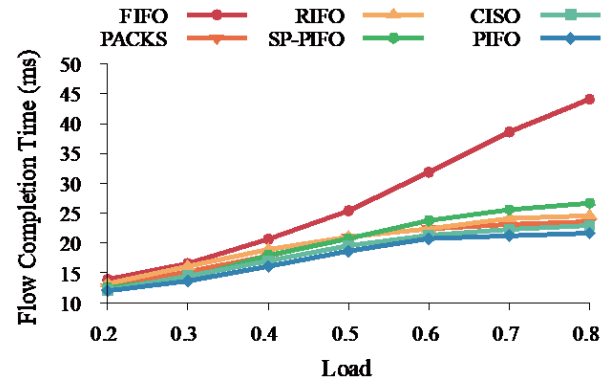
Fig. 3 FCT of short flows under different loads

图3 不同负载下短流的FCT

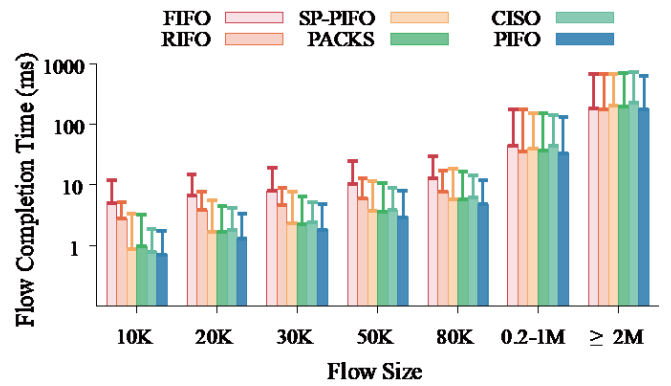
长流更关注吞吐量。图4(a)为长流的平均FCT。可以看出,在0.8的负载下,CISO的长流FCT相较RIFO降低约60%,相较SP-PIFO降低约25%,表明CISO的智能调度在优先保障短流的同时,通过其自适应的队列映射机制有效避免了长流“饿死”现象,保持了良好的公平性。图4(b)从流规模维度展示了不同流大小的FCT分布。可以看出,对于10 KB的短流,CISO的FCT比RIFO降低约70%;对于2 MB以上的大流,其FCT比SP-PIFO降低约30%。CISO在保障长流公平性方面的技术优势在于其动态性。与RIFO和SP-PIFO不同,CISO的控制平面周期性地重新评估rank分布,这意味着即使长流当前未被优先服务,一旦高优先级短流减少,K-means会重新计算质心,调整阈值,为长流腾出更高优先级的队列空间。这种自适应的资源再分配机制确保了长流不会永久“饥饿”,实现了整体的公平与高效。

3.2.3 BMv2 软件交换机中的带宽分配分析

图5为基线FIFO配置和在BMv2上运行的CISO调度程序的每个流量随时间变化的带宽分配。如图5(a)所示,默认FIFO在竞争流之间表现出平等的带宽共享。当Flow 1在 $t=0$ s开始传输时,其最初利用了全部的1 Gbps 瓶



(a) Average FCT of large flows
(a) 长流平均 FCT



(b) FCT at 70% load with different flow sizes
(b) 70% 负载时不同流大小的 FCT

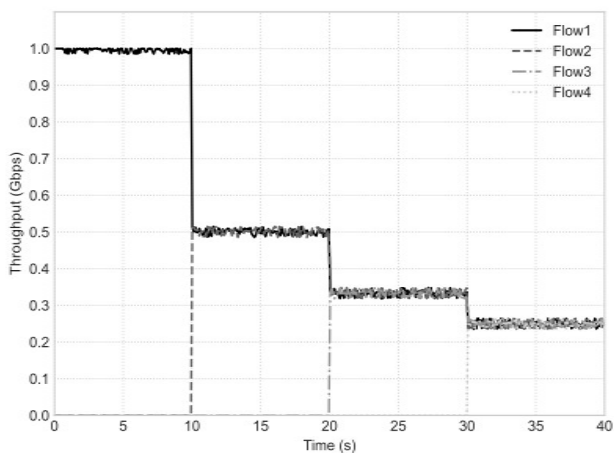
Fig. 4 FCT at different loads and flows

图4 不同负载与流大小下的FCT

颈容量。当Flow 2在 $t=10$ s到达时,带宽在两个流之间近似平均共享。随着后续流的到达,该模式持续进行,导致所有4个Flow在全部激活后最终以各约0.25 Gbps的带宽共享瓶颈容量。该行为证明了FIFO无法基于重要性或紧迫性来区分流量优先级。图5(b)展示了CISO基于优先级的带宽分配。当Flow 1单独处于活动状态时,其会利用全部瓶颈带宽。当优先级较高的Flow 2在 $t=10$ s到达时,CISO立即重新分配带宽以优先处理更高优先级的Flow,将Flow 1的吞吐量降至接近零,同时将几乎全部可用带宽分配给Flow 2。随着Flow 3和Flow 4的到达,这种抢占行为将继续存在,当前优先级最高的活动流主导链路容量。以上结果说明BMv2软件交换机能够正确利用控制平面提供的阈值,根据排名对传入数据包进行分类,并将其映射到适当的优先级队列,从而有效区分服务并按预期确定高优先级流量的处理级别。

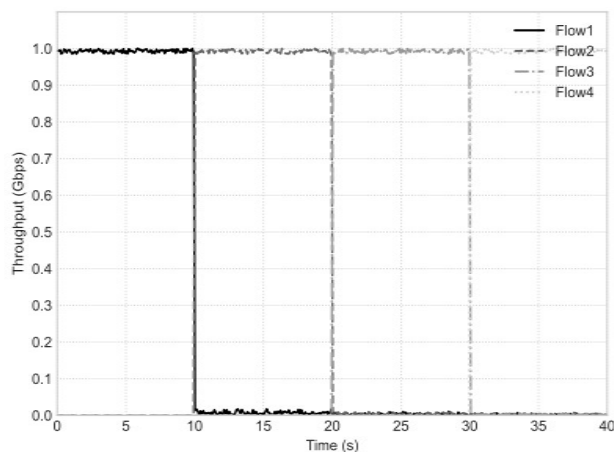
4 结语

本文提出基于K-means聚类的自适应包调度方法CISO,通过控制平面学习流量优先级分布并生成自适应阈值,在数据平面实现了高效的队列映射与调度决策。性能



(a) FIFO bandwidth-split

(a) FIFO带宽分配



(b) CISO bandwidth-split

(b) CISO带宽分配

Fig. 5 Bandwidth allocation for flows with increasing priority**图5 优先级递增流的带宽分配**

对比实验表明,CISO的调度逆序数相较PACKS和SP-PIFO等方法显著降低,同时在短流完成时间上达到与理想PIFO仅相差5%~9%的性能表现,表明该方法在保障调度精度和优化FCT方面具有明显优势。此外,CISO的数据平面仅需进行整数比较操作,资源消耗远低于PACKS等复杂方案,在实现高性能的同时保持了较高的资源利用效率。

然而,本研究仍存在一些不足。由于网络流量特性会随时间动态变化,控制平面的聚类周期和阈值更新策略需要进一步优化,以更好地适应实时流量模式。下一步研究将深入分析流量动态特性,探索自适应的聚类触发机制和阈值更新策略,在保持调度性能的同时进一步降低控制平面开销。

参考文献:

[1] WANG Z, YE J, LIN D, et al. Approximate and deployable shortest remaining processing time scheduler [J]. IEEE/ACM Transactions on Net-

working, 2022, 30(3): 1368–1381.

- [2] CHEN W, TIAN Y, YU X, et al. Enhancing fairness for approximate weighted fair queueing with a single queue [J]. IEEE/ACM Transactions on Networking, 2024, 32(5): 3901–3915.
- [3] NICKEL M, GOHRINGER D. A survey on architectures, hardware acceleration and challenges for in-network computing [J]. ACM Transactions on Reconfigurable Technology and Systems, 2024, 18(1): 1–34.
- [4] CHICA J C C, IMBACHI J C, VEGA J F B. Security in SDN: a comprehensive survey [J]. Journal of Network and Computer Applications, 2020, 159: 102595.
- [5] STUBBE H, GALLENMULLER S, SIMON M, et al. Exploring data plane updates on p4 switches with p4runtime [J]. Computer Communications, 2024, 225: 44–53.
- [6] WASEEM Q, ALSHAMRANI S S, Nisar K, et al. Future technology: software-defined network (SDN) forensic [J]. Symmetry, 2021, 13(5): 767.
- [7] YANG X R, ZENG L B, LIU Z P, et al. FastRMT: a high-speed data plane programmable system for microarchitecture innovation [J]. Chinese Journal of Computers, 2024, 47(2): 473–490.
杨翔瑞, 曾令斌, 刘忠沛, 等. FastRMT: 一种面向微体系结构创新的高速数据平面可编程系统 [J]. 计算机学报, 2024, 47(2): 473–490.
- [8] SAEED A, ZHAO Y, DUKKIPATI N, et al. Eiffel: efficient and flexible software packet scheduling [C]//16th USENIX Symposium on Networked Systems Design and Implementation, 2019: 17–32.
- [9] SIVARAMAN A, SUBRAMANIAN S, ALIZADEH M, et al. Programmable packet scheduling at line rate [C]//Proceedings of the 2016 ACM SIGCOMM Conference, 2016: 44–57.
- [10] ZHANG Z, CHEN S, YAO R, et al. Vpifo: virtualized packet scheduler for programmable hierarchical scheduling in high-speed networks [C]//Proceedings of the ACM SIGCOMM 2024 Conference, 2024: 983–999.
- [11] ALCOZ A G, VASS B, NAMYPAR P, et al. Everything matters in programmable packet scheduling [C]//22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25), 2025: 1467–1485.
- [12] LI Z, HU Y, TIAN L, et al. Packet rank-aware active queue management for programmable flow scheduling [J]. Computer Networks, 2023, 225: 109632.
- [13] ELBEDIWY M, PONTIKAKIS B, DAVID J P, et al. Enabling rank-based P4 programmable schedulers: requirements, implementation, and evaluation on BMv2 switches [J]. IEEE/ACM Transactions on Networking, 2024, 33(1): 299–310.
- [14] MOSTAFAEI H, PACUT M, SCHMID S. RIFO: pushing the efficiency of programmable packet schedulers [J]. IEEE Transactions on Networking, 2025, 1: 1295–1308.
- [15] YU Z, HU C, WU J, et al. Programmable packet scheduling with a single queue [C]//Proceedings of the 2021 ACM SIGCOMM 2021 Conference, 2021: 179–193.
- [16] SONG J, LIU J, GAO P, et al. Inversion impact of approximate pifo to start-time fair queueing [J]. Computer Networks, 2024, 240: 110164.
- [17] ALCOZ A G, DIETMULLER A, VANBEVER L. SP-PIFO: approximating push-in first-out behaviors using strict-priority queues [C]//17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), 2020: 59–76.
- [18] LIATIFIS A, SARIGIANNIDIS P, ARGYRIOU V, et al. Advancing SDN from openflow to p4: a survey [J]. ACM Computing Surveys, 2023, 55(9): 1–37.
- [19] CUGINI F, SCANO D, GIORGETTI A, et al. P4 programmability at the network edge: the braine approach [C]//2021 International Conference on Computer Communications and Networks (ICCCN), 2021: 1–9.

(责任编辑:尹晨茹)