计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(11): 2909−2924, 2026

可重配置数据中心的高效重配置与流量调度算法

赵佳欣[1,2] 王竞舟[1,2] 孙玉娥[3] 黄 河[1,2]
[1](苏州大学计算机科学与技术学院 江苏苏州 215006)
[2](数据智能与先进计算省高校重点实验室(苏州大学) 江苏苏州 215006)
[3](苏州大学轨道交通学院 江苏苏州 215137)
(20254227095@stu.suda.edu.cn)

Efficient Reconfiguring and Flow Scheduling Algorithms for Reconfigurable Data Centers

Zhao Jiaxin[1,2], Wang Jingzhou[1,2], Sun Yu'e[3], and Huang He[1,2]
[1] (*School of Computer Science and Technology*, *Soochow University*, *Suzhou*, *Jiangsu* 215006)
[2] (*Key Laboratory of Data Intelligence and Advanced Computing in Provincial Universities* (*Soochow University*), *Suzhou*, *Jiangsu* 215006)
[3] (*School of Rail Transportation*, *Soochow University*, *Suzhou*, *Jiangsu* 215137)

Abstract Thanks to the reconfigurable ability of optical switching technologies, data centers are able to adjust topologies at runtime to adapt to traffic dynamics generated by new technologies and applications. However, existing solutions for reconfigurable and traffic scheduling suffer from the high flexibility of configurations and traffic dynamics. Static solutions overlook practical constraints, fail to adapt to real-production environments. Meanwhile, dynamic solutions cannot guarantee global optimization, resulting in decrease in overall network performance. To bridge the gap, we first formalize the joint reconfiguration and traffic scheduling as the *k*-weight limited matching problem and propose an approximation algorithm for global updates based on priority queues and greedy iteration. Then, we divide the traffic dynamics into three different scenarios, each of which is addressed by an efficient local scheduling algorithm. To further improve network performance of both global update and local scheduling, we propose SYNC (static and dynamic network coordinator) through a periodic global optimization and graph structure-aware mechanism to achieve the trade-off between throughput and scheduling efficiency. We conduct simulation experiments with both real-world datasets and synthetic datasets generated by RMAT (recursive matrices). Experiments show that SYNC provides an efficient, stable and scalable scheduling solution for reconfigurable data center networks. Compared with existing solutions, SYNC reduces running time by 36.56%, and exhibits high stability at runtime.

Key words reconfigurable network; data center; optical circuit switch; graph matching; scheduling algorithm

摘 要 光交换技术得益于其可重配置特性,支持数据中心在运行时根据流量大小动态变化实时调整拓扑,可以有效应对新兴技术催生的全新流量模式。然而,配置的高度灵活性与流量大小的高度动态性是重配置与流量调度算法设计的瓶颈,现有的静态或动态算法方案难以兼顾网络性能与运行时延。基于此,提出一种静态-动态协同重配置与流量调度算法 SYNC(static and dynamic network coordinator)。将重配置

收稿日期: 2026-01-05; **修回日期**: 2026-05-20
基金项目: 国家自然科学基金项目(62332013); 江苏高校优势学科建设工程资助项目
This work was supported by the National Natural Science Foundation of China (62332013) and the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).
通信作者: 王竞舟(jzwang6972@suda.edu.cn)

与流量调度问题形式化为k-权重受限匹配问题,并基于优先队列和贪心迭代思想提出一套具有近似比保障的静态全局更新算法。详细分析可重配置数据中心中3类流量大小动态变化场景,针对性地为每类场景提出了高效动态局部调度算法。为进一步解决全局更新与局部调度间的平衡问题,通过周期性全局优化与图结构感知机制,实现系统性能与调度效率之间的权衡。基于真实生产环境数据集与RMAT(recursive matrices)生成的合成数据集的仿真实验结果表明,SYNC在保持竞争性吞吐率性能的同时达成了更短的运行时延,较已有算法降低运行时间达36.56%,同时表现出高度运行稳定性。

关键词 可重配置网络;数据中心;光路交换机;图匹配;调度算法

中图法分类号 TP393

DOI: 10.7544/issn1000-1239.202660004 **CSTR:** 32373.14.issn1000-1239.202660004

近年来,随着人工智能大模型、分布式机器学习等技术的快速发展,数据中心作为提供算力的关键基础设施,面临着流量规模迅速增长、流量大小动态变化加剧等全新挑战[1]。传统数据中心网络架构如Fat-Tree[2], Spine-Leaf[3]等,均由基于分组交换的传统交换机组成,并采用电信号传输数据,在传统业务中具有较好的适应性。然而,这些基于电交换的架构由于其固定拓扑特性,难以根据实时流量变化动态调整拓扑与设备间的连接方式[4],在新兴技术与应用催生的全新流量模式下面临链路过载和资源利用率低下等问题,从而导致用户服务质量下降[5-6]。此外这类网络架构不仅布线复杂,而且随着规模扩展布线难度和能耗呈指数级增长[7],制约了数据中心的可持续发展。因此,亟需设计一种更高效的数据中心架构方案来解决传统架构存在的共性问题。

新兴的光交换技术与设备为现有数据中心架构设计提供了新思路,其核心优势在于支持数据中心实时调整“配置”以适应流量大小动态变化。在本研究领域中,一个“配置”通常指光交换设备内部的端口连接方式及相应的网络拓扑。在各类光交换设备中,光路交换机(optical circuit switch, OCS)具有低重配置延迟[8]、高链路带宽[9]和低能耗等显著优势[10],并且与传统电交换网络兼容[11],因此基于OCS的可重配置数据中心(reconfigurable data center, RDC)已成为主流架构方案[12]。然而,尽管当前OCS已实现微秒级拓扑重配置[13],但受限于交换机内部高度灵活的端口连接方式与流量大小的动态变化特性,光交换网络重配置与相应流量调度方案的计算复杂性和控制开销仍然是限制其广泛应用的瓶颈。因此,改进光交换网络重配置与流量调度算法,优化网络整体性能,如提高系统吞吐率并降低重配置时延,对于提升OCS在实际数据中心环境中的适用性具有重要意义。

现有可重配置数据中心的重配置与流量调度算法主要分为静态和动态2类。静态算法在每次计算时均需根据全局流量、拓扑等信息重新生成全局配置,并决定如何调度流量,以确保全局最优性(如吞吐率最大化)。然而,静态算法时间复杂度较高,在大规模可重配置数据中心中难以实时响应流量大小动态变化,尤其在突发流量场景下,其滞后性可能导致网络资源未能得到充分利用。此外,静态算法的配置计算结果与当前配置方案独立,可能引入不必要的拓扑变更,从而增加系统开销[14]。相比之下,动态算法能够快速调整连接调度流量,以适应流量的短期波动。但由于其主要基于局部流量信息进行小范围的增量式优化,难以在全局范围内实现最优配置,长期运行可能导致网络性能下降,影响整体吞吐率和服务质量。

针对当前方案的缺陷,本文提出了一种面向全光架构可重配置数据中心的静态-动态协同重配置与流量调度算法SYNC(static and dynamic network coordinator),实现网络性能与控制开销之间的有效权衡。与现有仅支持单OCS调度机制[15]或多OCS独立调度机制[14,16]的研究不同,SYNC的显著优势在于支持多台OCS协同为同一通信需求分配光纤资源,从而进一步提升资源利用率。此外,该算法融合了全局更新与局部调度机制,基于周期性和图结构感知策略,在静态算法和动态算法间进行自适应切换,有效结合二者优势。其中静态算法可以实现近似最优地提升整体吞吐率,而动态算法降低频繁重配置带来的系统开销,进而实现对突发流量的快速响应。

本文工作的贡献包含以下4点:

1)提出了更贴合实际环境的新问题建模,即多OCS下的k-权重受限匹配问题,综合考虑光纤负载限制、流量协同传输等实际因素,并严格证明该问题的NP困难与不可近似性,揭示了其理论计算难度;

2)设计了一种基于优先队列与贪心迭代的静态全局更新算法，证明算法在光纤资源充足条件下具有1/2的近似比，并通过构造实例证明该近似比为紧界；

3)提出了一种改进的动态局部调度算法，实现对流量需求变化的快速响应，并通过引入阈值控制配置切换稳定性，实现微秒级调度延迟下的局部优化调整；

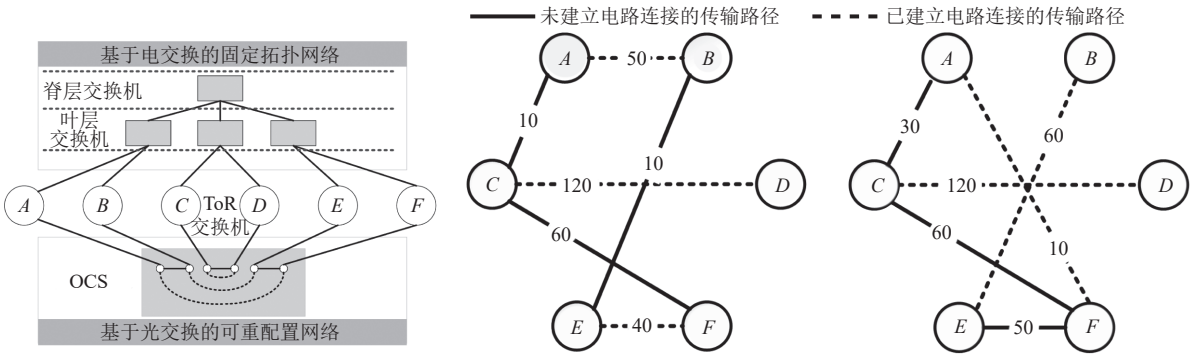
4)提出了静态-动态协同重配置与流量调度算法SYNC，该算法基于时间周期与图结构特征自适应选择静态或动态算法，在保证全局优化能力的同时兼顾局部调度效率。

1 研究背景与相关工作

本节将介绍OCS的原理以及SYNC算法的主流方案。

1.1 光路交换机原理

OCS是一种基于微机电系统(micro-electro-mechanical system, MEMS)技术的无阻塞光通信设备[17]，能够在无需进行光电转换的情况下，直接在输入和输出端口之间建立电路连接，并采用光信号承载大规模流量传输[18]，实现高速数据传输。以图1(a)为例，数据中心由6个机架(*A*~*F*)构成，每个机架均配备机架顶(top-of-rack, ToR)交换机。数据中心整体网络架构由2部分组成：上半部分为基于电交换的固定拓扑网络，采用脊-叶(spine-leaf)架构，其连接方式与网络拓扑在运行期间不可更改；下半部分由配备6个端口的OCS构成，具备微秒级的重配置能力[19]，实时调整端口间的电路连接以适应流量大小动态变化，以提高资源利用率和系统吞吐率。



(a) 数据中心架构及OCS光连接配置图 (b) 时刻T_1通信需求分布图 (c) 时刻T_2通信需求分布图

Fig. 1 Schematic diagram of OCS-based data center network architecture and dynamic reconfiguration

图1 基于OCS的数据中心网络架构与动态重配置示意图

如图1(b)所示，在时刻T_1机架对*A*—*B*, *C*—*D*, *C*—*F*分别存在50, 120, 60单位流量的通信需求，而*A*—*C*, *B*—*E*, *E*—*F*间通信需求的流量规模则相对较小，符合数据中心中流量大小分布不均的特点。受限于OCS的电路交换特性，在任一配置中每个ToR交换机最多仅能建立1条连接。为提升整体吞吐效率，系统优先为*A*—*B*, *C*—*D*, *E*—*F*这3对流量规模较大的ToR交换机之间建立电路连接，其传输路径在图1(a)(b)中分别以实线和虚线表示。至时刻T_2，通信需求流量发生明显变化，如图1(c)所示。其中，*C*—*D*和*C*—*F*依旧维持120与60单位流量的通信需求，同时*B*—*E*(60单位)、*E*—*F*(50单位)、*A*—*C*(30单位)与*A*—*F*(10单位)的通信需求也显著上升。为应对流量大小的动态变化，OCS可在数微秒内完成重配置：断开原有的*A*—*B*和*E*—*F*连接，并将释放的端口资源重新分配，用于构建*C*—*D*, *A*—*F*, *B*—*E*的电路连接。该重配置结果在图1(a)(c)中以虚线表示。

通过对比图1显示的时刻T_1与T_2的光网络配置，OCS的可重配置特性使网络能够在不同时间步灵活选择当前流量规模较大的通信需求传输，如从T_1的*A*—*B*和*E*—*F*切换至T_2的*A*—*F*和*B*—*E*，从而缓解固定拓扑下的资源分配瓶颈。OCS微秒级的重配置延迟进一步保障了网络对突发流量模式的快速响应能力，显著提升资源利用率和用户服务质量。

1.2 研究现状分析

近年来，全光架构可重配置数据中心的重配置与流量调度算法的研究取得了显著进展，主要围绕静态算法与动态算法展开，并有部分研究尝试结合二者优势，提出混合算法方案。由于OCS的电路交换特性使得在任一配置中，每个ToR交换机最多仅能建立1条连接，因此OCS的“配置”符合图论中“匹配”的定义，故当前多数重配置算法均基于匹配算法

设计。例如，通过将 ToR 交换机抽象为图中节点，ToR 交换机之间的通信需求抽象为图中的边，边权重对应通信需求的流量大小，如何调整 OCS 配置最大化吞吐率即可抽象为最大边权重匹配问题。

在静态算法领域，早期研究以单一 OCS 为基础，使用 Edmonds 提出的 Blossom 算法[20]求解最大边权重图匹配问题[15]，且多年来相关问题算法的近似比和运行时间都在不断改进[21]。但单个 OCS 的设置难以满足现代数据中心的高吞吐率需求，为此 Farrington 等人[22]提出了 Helios 解决方案，对于每个 OCS 贪心地选择具有最大流量的连接配置，即具有最大边权重的匹配。但 Helios 算法缺乏近似比保证且仅考虑均匀光纤容量，实际运行的吞吐率性能较差。Hanauer 等人[16]进一步将问题形式化为 k-不相交匹配（k-disjoint matching，k-DM）问题，对每个 OCS 重复调用 Blossom 算法，并证明其近似比不超过 7/9。然而，该方法未考虑光纤容量的限制，导致某些流量较高的通信需求仅被单个 OCS 选中，无法将流量完全传输，降低系统的整体吞吐率。针对这一问题，Wang 等人[23]提出的 Leaf 算法创新性地将问题建模为 k-权重受限匹配问题，并采用改进的贪心迭代算法实现了 1/2 近似比，在多 OCS 协作调度及光纤容量约束方面取得了突破。然而，以上所有方案在每次调度时都需对全局配置及流量调度方案进行重新计算，未能充分利用部分流量变化对拓扑结构影响有限的特性。例如，当已建立连接的 ToR 交换机之间流量增加时，OCS 的最优配置可能保持不变，而此类方案仍然执行全局更新，增加了计算开销。此外 OCS 的频繁重配置可能会中断或新建大量连接，严重影响网络稳定性和数据传输的连续性。

动态算法的研究近年来备受关注，其核心在于利用流量的时空局部性特征进行增量优化。Hanauer 等人[14]将问题从 k-DM 问题转换为部分 k-染色问题，并提出 2 种动态算法。dyn-greedy 对权重变化的边进行局部调整，通过递归尝试和颜色替换优化匹配，并引入随机化参数来随机选择颜色子集。由于该算法采用局部贪心更新，时间效率较高。dyn-kEC 是基于静态 kEC 算法的动态版本，Hanauer 等人[16]在权重变化时，尝试对受影响边重新着色，优先保留高权重边；当待处理边无法直接分配颜色时，设计了颜色旋转方法来释放可用颜色。但受 kEC 结构限制，算法仍需遍历部分匹配，在大规模拓扑更新场景下计算负担较大。此外，Hanauer 等人[16]还提出批量动态算法 batch-greedy 和 batch-NC，采用固定时间窗口聚合多个更新请求，在窗口到达后执行统一优化，从而减少计算频率。但该研究未提供如何根据流量动态性自适应调整窗口大小的机制，在流量更新较为突发或剧烈变化的场景下，窗口过大会导致调度滞后，而窗口过小则会影响批量优化的效果。此外，这些动态算法主要专注于理论研究，并未充分考虑光纤数量以及容量的限制，导致实际吞吐率较低。

混合算法方案旨在结合静态算法的全局优化能力和动态算法的局部快速调整特性，但目前的相关工作较少。Hanauer 等人[14]提出的 hybrid-kEC 算法和 hybrid-greedy 算法创新性地引入自适应决策机制，即当批量更新比例较低时，采用动态匹配优化算法，否则切换到静态 kEC 以重新计算匹配。然而，这类混合方法仍存在静态与动态算法协同效率不足的缺陷，当流量变化模式极端化时，算法可能会退化为纯静态或纯动态方案，未能充分发挥混合算法的优势。

总体而言，现有的重配置与流量调度算法研究在算法效率与优化效果的平衡方面仍存在显著缺陷。静态算法的时间复杂度在面对实时流量调度时仍会产生毫秒级延迟，难以充分发挥 OCS 微秒级重构的硬件优势。动态算法虽通过增量更新将单次处理时间降至微秒量级，但长期运行会影响全局性能，如导致吞吐率降低。混合算法目前的研究成果还较少，尽管现存方案一定程度上融合了静态全局优化与动态局部调度的优势，但在极端流量变化情况下，仍然可能面临静态算法和动态算法各自的局限。此外，现有工作大多基于无限光纤容量假设，忽略了实际数据中心中带宽分配不均等因素，限制了算法的适用性。

2 重配置与流量调度问题建模

SYNC 涉及的变量与参数符号及其具体含义详见表 1。

2.1 全光架构可重配置数据中心模型

将全光架构可重配置数据中心 ToR 交换机之间的通信需求建模为简单、无向、具有随时间变化的边权重的图 $G^t=\{V,E^t,W^t\}$。顶点集合 $V=\{v_1,v_2,\cdots,v_n\}$ 表示 ToR 交换机集合，其中 $n=|V|$ 表示 ToR 交换机数量。在数据中心网络中，任意 2 台 ToR 交换机之间都可能存在通信需求，边集合 $E^t=\{e_1,e_2,\cdots,e_m\}$ 表示时刻 t 的通信需求集合，其中 $m^t=|E^t|$ 代表在时刻 t 的通信需求总数。时间相关的边权重函数 $W^t\colon E^t\to N_0$ 用于表示在时刻 t 每个通信需求对应的数据流量大小。设 I 表示 OCS 集合，记作 $I=\{i_1,i_2,\cdots,i_k\}$，其中 k 代表

Table 1 List of Nomenclature

表 1 符号表

符号	含义
$G^t=\{V,E^t,W^t\}$	在时刻t具有时间相关边权重的图
V	ToR 交换机的集合
$n=\|V\|$	ToR 交换机的数量
E^t	在时刻t的通信需求集合
$m^t=\|E^t\|$	在时刻t的通信需求总数
$W^t: E^t\to N_0$	在时刻t时每个通信需求对应的数据流量大小
I	OCS 集合
$k=\|I\|$	OCS 的数量
M_i^t	OCS_i在时刻t的连接配置
L_i	OCS_i配备的光纤集合
$c(l)$	光纤l的容量
$N^t(e)$	边e在时刻t的邻边集合
$N_i^t(e)$	边e的邻边中在时刻t被OCS_i的匹配M_i^t选中的边
$x_{e,i,l}^t$	二元变量，边e在时刻t是否被分配至OCS_i的光纤l
$y_{e,i,l}^t$	流量调度变量，边e在时刻t时在OCS_i的光纤l上分配的流量大小
MAX	一个足够大的常数，表示流量的上限
$D(M_i^t)$	OCS_i在时刻t的总实际吞吐量

OCS 的数量。OCS_i在时刻t时对 ToR 交换机之间的连接配置根据电路交换的特性可被建模为图G^t中的 1 个匹配M_i^t。每台 OCS 均配备有 1 组光纤，记作L_i。每条光纤l的容量为$c(l)$，光纤l上传输的流量不得超过其容量限制。在时刻t时边$e=\{u,v\}$的邻边集合记作$N^t(e)$，定义为：$N^t(e)=\{e'\in E^t\backslash\{e\}\,|\,e'\cap\{u,v\}\neq\varnothing\}$。邻边中被$OCS_i$在时刻$t$的匹配$M_i^t$选中的边记作$N_i^t(e)$，其中，$N_i^t(e)=N^t(e)\cap M_i^t$。

2.2 问题形式化

重配置与流量调度问题涉及 2 套独立变量，定义如下：重配置变量$x_{e,i,l}^t$为二元变量，表示在时刻t时，通信需求e是否由OCS_i的光纤l传输，若被分配，则$x_{e,i,l}^t=1$；否则，$x_{e,i,l}^t=0$。流量调度变量$y_{e,i,l}^t$表示在时刻t时，通信需求e在OCS_i的光纤l上分配的流量大小，取值范围为$0\leqslant y_{e,i,l}^t\leqslant\min(W^t(e),c(l))$，其中$W^t(e)$为时刻$t$时边$e$的流量需求，$c(l)$为光纤$l$的容量。$x_{e,i,l}^t$和$y_{e,i,l}^t$之间存在关系：$y_{e,i,l}^t\leqslant x_{e,i,l}^t\times MAX$，其中$MAX$是一个足够大的常数。当$x_{e,i,l}^t=1$时，$y_{e,i,l}^t$即可取正值，代表该光纤可以进行流量传输；当$x_{e,i,l}^t=0$时，$y_{e,i,l}^t=0$，表示通信需求$e$的流量不会被分配给$OCS_i$的光纤$l$。每个 OCS 在时刻$t$的总实际吞吐量表示为$D(M_i^t)=\sum_{e\in M_i^t}\sum_{l\in L_i}y_{e,i,l}^t$，表示该配置在时刻$t$所承载的总流量。优化目标是计算$k$个 OCS 的配置，使得时刻$t$的总吞吐量$\sum_{i=1}^{k}D(M_i^t)$最大化。此外，本问题包括如下约束。

1）负载约束。在任意时刻，每个 OCS 所使用的每条光纤所承载的总流量不能超过其负载上限：

$$0\leqslant y_{e,i,l}^t\leqslant c(l),\ \forall t,e\in E^t,i\in I,l\in L_i\text{。}\tag{1}$$

2）流量需求约束。在任意时刻，对于每个通信需求，所有 OCS 所承载的流量总和不能超过该通信需求自身的流量大小：

$$\sum_{i\in I}\sum_{l\in L_i}y_{e,i,l}^t\leqslant W^t(e),\ \forall t,e\in E^t\text{。}\tag{2}$$

3）电路约束。由于 OCS 受电路交换的特性约束，在任意时刻，每个 ToR 交换机在每个配置中最多只能连接到 1 个其他 ToR 交换机：

$$\sum_{e\in E:v\in e}x_{e,i,l}^t\leqslant 1,\ \forall t,i\in I,l\in L_i,v\in V\text{。}\tag{3}$$

4）稳定性约束。为了保持稳定性，当前时刻 OCS 的配置与上一时刻 OCS 的配置不能差距过大。因此对每一个 OCS 计算当前时刻配置与上一时刻配置发生变化的连接数量并进行求和，该数值不能超过稳定性阈值β：

$$\sum_{e\in E}|x_{e,i,l}^t-x_{e,i,l}^{t-1}|\leqslant\beta,\ \forall t\geqslant 1,i\in I,l\in L_i\text{。}\tag{4}$$

基于上述分析，本文将全光架构可重配置数据中心重配置与流量调度问题形式化为

$$\max\sum_{i=1}^{k}D(M_i^t),\ \forall t\text{。}$$

$$\text{s.t.}\begin{cases}D(M_i^t)=\sum_{e,l}y_{e,i,l}^t,\ \forall t,M_i^t,i\text{。}\\ y_{e,i,l}^t\leqslant c(l),\ \forall t,e\in E^t,i\in I,l\in L_i\text{。}\\ y_{e,i,l}^t\geqslant 0,\ \forall t,e\in E^t,i\in I,l\in L_i\text{。}\\ y_{e,i,l}^t\leqslant x_{e,i,l}^t\times MAX,\ \forall t,e\in E^t,i\in I,l\in L_i\text{。}\\ \sum_{i\in I}\sum_{l\in L_i}y_{e,i,l}^t\leqslant W^t(e),\ \forall t,e\in E^t\text{。}\\ \sum_{e\in E:v\in e}x_{e,i,l}^t\leqslant 1,\ \forall t,i\in I,l\in L_i,v\in V\text{。}\\ \sum_{e\in E}\left|x_{e,i,l}^t-x_{e,i,l}^{t-1}\right|\leqslant\beta,\ \forall t\geqslant 1,i\in I,l\in L_i\text{。}\end{cases}$$

2.3 问题复杂性分析

与以往将问题建模为k-DM 问题[16]的研究不同，本文考虑到实际光纤容量的限制，允许多个 OCS 共同传输同一对 ToR 交换机间的通信需求，即图中的 1 条边允许被多个匹配包含，而 1 个匹配的总权重被严格限制，因此该问题被定义为k-权重受限匹配问题。本节将系统分析该问题的计算复杂性，并形式化地

证明其不仅是NP困难的,而且具有不可近似性。

定理 1. k-权重受限匹配问题在$k \geqslant 3$时是NP困难的。

证明. 设图中所有边的权重均为正整数,光纤容量为有限值,目标是在给定OCS的数量k和光纤容量限制下,选择k个不相交的匹配,使得所选边的总权重最大。为证明该问题为NP困难问题,构造如下归约:假设所有光纤容量均大于等于图中任意边的最大权重,即每条边的通信需求都可以被任意一根光纤完全承载。在这种情形下,容量约束不再起作用,问题退化为k-不相交匹配问题。而根据已有研究[15-16],当$k=1$时,此问题为最大边权重图匹配问题,是多项式时间可解的;当$k \geqslant 3$时,k-不相交匹配问题是NP困难的。因此,可推出k-权重受限匹配问题在$k \geqslant 3$时是NP困难的。

进一步设想,若存在一个算法能够在有或无光纤容量约束的情况下均可求解出k-权重受限匹配问题的最优解,则该算法同样可以用于上述退化情形,从而间接解决已知的NP困难问题。因此可以得出结论:k-权重受限匹配问题在$k \geqslant 3$时为NP困难成立。

证毕.

定理 2. 对于任意$\epsilon \in [0, 1/m^t)$,其中m^t为图中边的数量,以$(1-\epsilon)$的近似比求解k-权重受限匹配问题是NP困难的。

在论证其不可近似性之前,先引入以下2个相关引理。

引理 1. 维津定理(Vizing theorem)指出,对于一个简单图,用δ表示图的最大度,则该图的边着色数等于δ或$\delta+1$[24]。

引理 2. Holyer定理指出,判定一个3次图(即所有顶点度数均为3)的色度指数是3还是4是NP完全问题[25]。

证明. 假设存在一个近似算法,对于任意$\epsilon \in [0, 1/m^t)$,该算法可输出一个近似比至少为$(1-\epsilon)$的解。构造一个3次图,设所有边的权重均为1,光纤容量为无限大,即忽略容量限制,取$k=3$,即尝试使用3个不相交匹配覆盖尽可能多的边。若该3次图的色度指数为3,即边可被完全划分为3个不相交的匹配组,最优解可覆盖全部m^t条边;若色度指数为4,则无法用3个匹配覆盖全部边,最多仅能覆盖m^t-1条边。假设$\epsilon = 1/m^t$,此时$(1-\epsilon)m^t = m^t - 1$,对于色度指数为3的图,近似算法应返回至少$m^t-1$条边;而图的色度指数为4时,最优解为$m^t-1$。因此,近似比严格大于$(1-1/m^t)$的算法若返回$m^t$条边,则说明图的色度指数为3;若返回少于$m^t$条边,则说明色度指数为4。由此即可判定3次图的色度指数,这与Holyer定理相矛盾。

证毕.

综上,k-权重受限匹配问题不仅在计算复杂性上是NP困难的,而且在近似计算层面也存在障碍。即使允许一定误差,也无法设计出近似比严格大于$(1-1/m^t)$的多项式时间算法。这进一步说明相较于单个OCS的最大匹配问题或不考虑光纤容量限制的k-不相交匹配问题,针对本问题设计一种时间复杂度低且近似比有界的近似算法具有很高的复杂性与理论研究价值。

3 流量调度算法的设计与实现

SYNC算法的核心思想在于“静态全局更新”与“动态局部调度”的结合。相较于已有工作,静态算法设计难点在于如何解决“如何分配同一条边的权重给不同匹配”;动态算法设计难点在于“如何快速处理权重变化的边”;而最终整体算法的设计难点在于“如何确定静态全局更新算法的触发条件”。针对以上3点,分别设计了基于贪心迭代的近似算法、基于权重比较的置换操作、基于周期与图结构感知的触发方法。

3.1 静态全局更新算法

本节提出了一种基于优先队列的贪心迭代算法以实现全局更新,具体流程如算法1所示。其核心思想是:“大流量需求优先匹配,高容量光纤优先分配”。具体而言,算法首先将所有边$e \in E^t$放入优先队列Q_e,该队列根据边的权重$W^t(e)$以降序排序(行①)。随后,对每台OCS独立进行配置:将其配备的光纤按容量降序排列,并依次从Q_e中检索最大权重的合法边(即与当前OCS的匹配集合M_i^t无顶点冲突的边)进行匹配(行②~⑥)。一旦确定合法边,系统即根据光纤的最大承载能力分配流量(行⑦~⑩);若该边仍有未满足的剩余流量,则更新其权重并重新插回Q_e,以供后续可用光纤继续调度(行⑪~⑬)。当某台OCS无法从队列中检索到任何合法边时,算法即可提前终止该OCS的匹配过程(行⑭~⑰)。

通过该贪心迭代策略,算法能够依据流量需求与光纤容量,构建多组匹配方案完成流量分配,为数据中心网络的静态配置提供高效、可靠的结果。

算法 1. 静态全局更新算法。

输入:图$G^t = \{V, E^t, W^t\}$,OCS集合I,每个OCS_i的光纤集合L_i,容量函数$c(l)$;

输出：OCS 配置集合 M_i^t，重配置决策 $x_{e,i,l}^t$，流量调度决策 $y_{e,i,l}^t$。

① 初始化边的优先队列 Q_e，按边权重 $W^t(e)$ 降序排列；
② for 每个 $OCS_i \in I$ do
③ 初始化匹配集合 $M_i^t \leftarrow \varnothing$；
④ 初始化光纤的优先队列 Q_l 按容量 $c(l)$ 降序排列；
⑤ for Q_l 中的每根光纤 l do
⑥ 在 Q_e 中检索权重最大且合法的边 e；
⑦ if 成功找到合法边 e then
⑧ $M_i^t \leftarrow M_i^t \cup \{e\}$；
⑨ $x_{e,i,l}^t, y_{e,i,l}^t \leftarrow 1, \min(c(l), W^t(e))$；
⑩ $W^t(e) \leftarrow \max(0, W^t(e) - y_{e,i,l}^t)$；
⑪ if $W^t(e) > 0$ then
⑫ 将边 e 重新插入优先队列 Q_e；
⑬ end if
⑭ else
⑮ break;
⑯ end if
⑰ end for
⑱ end for

3.2 动态局部调度算法

本节提出了一种改进的动态局部调度算法，核心思想在于基于高效的边权重比较对匹配中的边进行快速置换，具体流程如算法 2 所示。首先初始化变量 $x_{e,i,l}^t$，$y_{e,i,l}^t$，M_i^t，使其继承前一时刻的分配状态。此外，定义变量 *changeCount* 为 0 用于记录当前匹配相对于前一次匹配的调整次数（行①），并计算待处理边的流量变化量 $\Delta W^t(e) = W^t(e) - W^{t-1}(e)$（行②）。接下来处理以下 3 类边权重变化情况：1）已分配边权重增加（行⑤⑥）；2）未分配边权重增加（行⑦~⑱）；3）已分配边权重减少（行⑲~㉔）。未分配边权重减少的情形则无需处理。

当边 e 的流量需求增加时，算法首先采取开销最小的贪心策略：优先利用该边当前已分配光纤的剩余可用容量来满足增量流量。若流量仍未被完全满足，算法会尝试在 OCS 集合中为其寻找新的空闲光纤。

为了进一步优化资源利用率，算法引入了 2 种基于权重比较的拓扑置换操作：SwapIn 用于处理未分配边权重增加的情况。当边 e 与 OCS_i 的当前匹配 M_i^t 存在冲突时，如果存在 $W^t(e) > W^t(N_i^t(e))$，那么 SwapIn 会将边 e 加入匹配 M_i^t，并将 $N_i^t(e)$ 中的边从匹配 M_i^t 中移除，然后再对被移除的边执行 SwapIn 算法直至达到递归深度 α。SwapOut 则用于处理已分配边权重减少的情况。尝试在 $N^t(e)$ 中寻找最多 2 条不相邻且未加入任何匹配的边 e_1 和 e_2，要求：1）e_1 和 e_2 与 e 属于的匹配 M_i^t 中的所有边均不相交；2）OCS_i 至少存在 1 条未分配光纤；3）满足 $W^t(e) < W^t(e_1) + W^t(e_2)$。此时 SwapOut 将边 e_1 和 e_2 加入匹配 M_i^t，并将 e 从匹配 M_i^t 中删除，重复执行算法直至达到递归深度 α。

算法 2. 动态局部调度算法。

输入：图 $G^t = \{V, E^t, W^t\}$，OCS 集合 I，光纤集合 L_i，容量函数 $c(l)$，上一时刻的重配置决策 $x_{e,i,l}^{t-1}$、流量调度决策 $y_{e,i,l}^{t-1}$、匹配集合 M_i^{t-1}，递归深度 α，稳定性阈值 β；

输出：新的重配置决策 $x_{e,i,l}^t$，流量调度决策 $y_{e,i,l}^t$，匹配集合 M_i^t。

① 初始化 $x_{e,i,l}^t$，$y_{e,i,l}^t$，M_i^t，$changeCount \leftarrow x_{e,i,l}^{t-1}$，$y_{e,i,l}^{t-1}$，$M_i^{t-1}$，0；
② 计算流量变化 $\Delta W^t(e) = W^t(e) - W^{t-1}(e)$；
③ for $\Delta W^t(e) \neq 0$ 的每条边 $e \in E^t$ do
④ if $changeCount \geqslant \beta$ then
⑤ break;
⑥ end if
⑦ if $\Delta W^t(e) > 0$ then
⑧ 若边 e 已分配光纤，则优先贪心利用其所属光纤的剩余容量分配 $\Delta W^t(e)$；
⑨ if $\Delta W^t(e) > 0$ then
⑩ 尝试在 I 中为 e 分配新的光纤并更新决策，直至流量满足或达到 β；
⑪ end if
⑫ if $\Delta W^t(e) > 0$ then
⑬ for 每一台 $OCS_i \in I$ do
⑭ if $changeCount \geqslant \beta$ then
⑮ break;
⑯ end if
⑰ 找到所有邻边 $e' \in N_i^t(e)$；
⑱ if $\sum_{e'} W^t(e') < W^t(e)$ then
⑲ 执行 SwapIn，所有 e' 递归调用本算法；
⑳ end if
㉑ end for
㉒ end if
㉓ else if $\Delta W^t(e) < 0$ then
㉔ 找到未分配且可加入 M_i^t 的邻边 $e_1, e_2 \in N(e)$；
㉕ if $W^t(e_1) + W^t(e_2) > W^t(e)$ then
㉖ 执行 SwapOut，e 递归调用本算法；

㉗ end if

㉘ end if

㉙ end for

通过上述策略,算法能够动态适应流量变化,快速执行 OCS 资源的有效分配。

3.3 静态-动态协同重配置与流量调度算法

本节提出了一种静态-动态协同重配置与流量调度算法 SYNC,兼顾全局优化与局部快速响应,流程图如图 2 所示。算法 SYNC 如算法 3 所示。

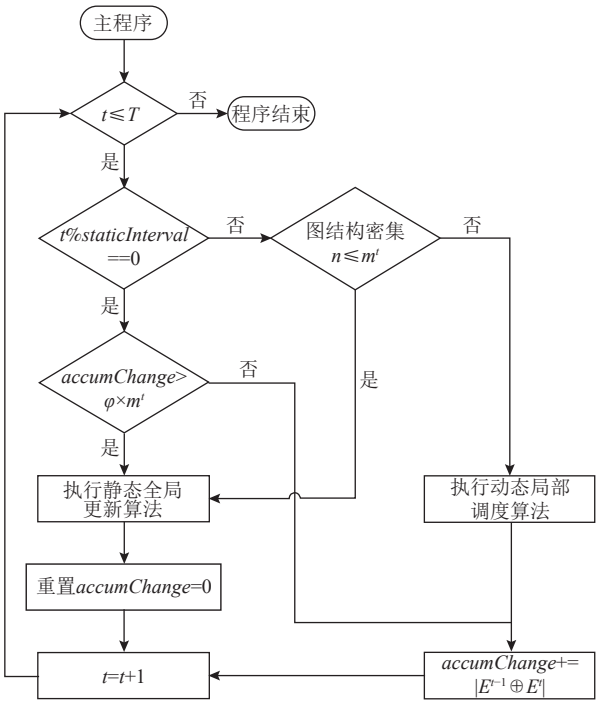


Fig. 2 Flow chart of SYNC

图 2 SYNC 流程图

SYNC 周期性地触发全局静态更新,用于重配置当前所有 OCS,并获得近似最优的流量调度结果。在静态算法执行周期内,若检测到边的权重发生变化,如通信需求增加或减少,则触发局部动态调度机制,根据 ToR 间的通信需求流量变化情况快速调整局部匹配,以维持系统的稳定性和服务连续性。然而在实际情况中,1 个时间步内可能同时新增大量通信需求,也可能仅涉及少量流量更新。动态算法的优势在于其面向单条边的调度时仅考虑局部情况,故运行速度远快于静态算法。但在高负载场景下,若对每条边都逐一使用动态算法可能适得其反,导致整体运行时间反而超过静态算法。因此,本研究在固定全局更新周期的基础上进一步引入结构感知机制。当某一时刻的边数超过节点数,即图结构趋于密集,说明当前流量更新范围较大,此时选择静态调度策略以提升整体效率;反之,当边数少于节点数时,仍使用动态算法以降低运行时间开销。同时,由于不同周期通信需求变化程度不一,因此引入变量 *accumChange* 记录一个周期内新增或中断的通信需求,若 *accumChange* 小于一定阈值,则说明变化程度较低,无需执行全局算法。

每个时间步中,SYNC 首先判断当前时间是否为执行静态算法的周期,若是,则判断上一周期变化的通信需求总数是否超出阈值,若超过则执行静态全局更新算法并将 *accumChange* 置 0,否则继续累加变动的连接数量。若当前并非执行静态算法的周期,则判断当前节点数是否小于等于边数。若是,则执行全局静态更新算法;否则执行动态算法,并更新 *accumChange* 的值。时间步计数器 t 在每个时间步后自动递增,从而实现静态与动态调度模式的切换。

算法 3. SYNC。

输入:图 $G^t=\{V,E^t,W^t\}$,OCS 集合 I,光纤集合 L_i,容量函数 $c(l)$,上一时刻的重配置决策 $x_{e,i,l}^{t-1}$、流量调度决策 $y_{e,i,l}^{t-1}$、匹配集合 M_i^{t-1},递归深度 α,稳定性阈值 β,静态调度周期 *staticInterval*,终止时间 T;

输出:新的重配置决策 $x_{e,i,l}^{t}$,流量调度决策 $y_{e,i,l}^{t}$,匹配集合 M_i^t。

① 初始化 $t, accumChange \leftarrow 0, m^t$;

② while $t \leqslant T$ do

③ if t % *staticInterval*== 0 then

④ if $accumChange > \varphi \times m^t$ then

⑤ 执行静态全局更新算法;

⑥ $accumChange \leftarrow 0$;

⑦ else

⑧ $accumChange \leftarrow accumChange + |E^{t-1} \oplus E^t|$;

⑨ end if

⑩ else if $n \leqslant m^t$ then

⑪ 执行静态全局更新算法;

⑫ $accumChange \leftarrow 0$;

⑬ else

⑭ 执行动态局部调度算法;

⑮ $accumChange \leftarrow accumChange + |E^{t-1} \oplus E^t|$;

⑯ end if

⑰ $t \leftarrow t+1$;

⑱ end while

通过该静态-动态协同策略,能够在保障数据中心网络长期整体性能的同时,有效应对流量动态变化带来的即时需求波动,在全局最优性与局部响应速度之间实现良好的平衡。

3.4 性能分析

本节对算法的时间复杂度与近似比进行分析。

定理 3. 静态全局更新算法时间复杂度为$O(m^t\log m^t+k(r\log r+m^t r^2))$，其中$k$是 OCS 的数量，$m^t$是通信需求（边）的数量，$n$是 ToR 交换机（顶点）的数量，$r$是每台 OCS 拥有的光纤数量。

证明. Leaf-PQ 算法首先将边集E^t按照权重$W^t(e)$进行排序，时间复杂度为$O(m^t\log m^t)$，其中$m^t=|E^t|$。该算法共进行k次迭代，每次迭代由排序光纤、为光纤选择匹配边，以及判断候选边是否与当前匹配集M_i^t冲突 3 部分组成。对于每台$OCS_i\in I$，其光纤集合L_i的大小记作$r_i=|L_i|$，光纤的排序复杂度为$O(r_i\log r_i)$，若每台 OCS 拥有至多r根光纤，即$r_i\leqslant r$，则光纤排序复杂度可统一表示为$O(r_i\log r_i)$。在分配阶段，对于每台OCS_i，算法尝试将每根光纤$l\in L_i$分配给 1 条和当前匹配未冲突的边。在最坏情况下，每根光纤需遍历整个边的优先队列Q_e，并检查候选边与当前匹配集合M_i^t中的所有边是否存在冲突。检查冲突最多需遍历r条边，每根光纤最多进行m^t次匹配尝试，故分配复杂度为$O(m^t r^2)$。因此，整个静态调度算法的总时间复杂度为$O(m^t\log m^t+k(r\log r+m^t r^2))$。在实际应用中，$r\ll m^t$，$k$和$r$通常为较小常数，因此算法的主导项为$O(km^t r^2)$，呈现多项式时间复杂度。

证毕.

定理 4. 动态局部调度算法在处理不同类型的边权重变化时，其时间复杂度如下：处理一条未分配边权重增加的时间复杂度为$O(k(r(1+\log r)+n\times 2^\alpha))$；处理一条已分配边权重增加的时间复杂度为$O(k(1+\log k+r(2+\log r)+n\times 2^\alpha))$；处理一条已分配边权重减少的时间复杂度为$O(k(n^2r^2+r(1+\log r)+n))$。其中，$k$表示 OCS 的数量，$n$表示 ToR 交换机的数量，$r$表示每台 OCS 所拥有的光纤数量上限，$\alpha$表示递归深度上限。

证明. Dynamic 算法基于边权重变化处理边的流量调度，主要分为 3 类情形。

1）未分配边权重增加。算法首先尝试将边e添加进某个 OCS 的匹配中，需遍历k个 OCS，并在每个 OCS 中查找容量最大的光纤，时间复杂度为$O(kr(1+\log r))$。若所有 OCS 的当前匹配均与e存在冲突，则需调用$SwapIn(e,i)$，遍历所有 OCS 与e的邻边集合并比较权重。此外，$SwapIn(e,i)$会递归移除邻边并尝试重新调度这些边，每轮最多触发 2 条边递归，深度限制为α，因此该部分的时间复杂度为$O(kn\times 2^\alpha)$。故未分配边权重增加的时间复杂度为$O(k(r(1+\log r)+n\times 2^\alpha))$。

2）已分配边权重增加。该情形下，算法需遍历边e当前分配的所有(i,l)组合，根据光纤剩余容量对其排序并尽量追加分配新增流量，每条边最多涉及k个 OCS 和每个 OCS 的 1 根光纤，时间复杂度为$O(kr+k\log k+k)$。若现有光纤无法满足新增流量，则会尝试新增分配光纤，此时处理流程等价于边权重增加的情形，因此最终时间复杂度为$O(k(1+\log k+r(2+\log r)+n\times 2^\alpha))$。

3）已分配边权重减少。算法将尝试执行$SwapOut(e)$操作。对于每个 OCS，在$N_i^t(e)$中寻找 2 条不相邻、未分配且可加入匹配M_i^t的边对，时间复杂度为$O(kn^2r^2)$。此外，算法需要尝试重新为e分配光纤，此时设置$\alpha=0$，故已分配边权重减少的时间复杂度为$O(k(n^2r^2+r(1+\log r)+n))$。

因此，Dynamic 算法在应对不同类型的边权重变化时，均能在多项式时间内完成调度。由于k和r通常为较小常数，且β限制了最大处理次数，故 Dynamic 算法能快速响应一条边的流量变化情况。

证毕.

定理 5. 当所有光纤的容量均大于所有边的最大权重时，静态全局更新算法的近似比为 1/2。

证明. 设M为 Leaf-PQ 算法输出的匹配，M^*为最优匹配。由于该算法按照边权重从大到小的顺序构造M，因此每次选择的边都是当前未匹配边中权重最大的那 1 条。考虑任意 1 条未被选入M的边$e^*\in M^*\backslash M$。由于e^*未被加入M，说明其冲突集中已存在某条边$e\in M$，且满足$W^t(e^*)\leqslant W^t(e)$，即e^*被权重更大的冲突边e所排斥。由于每条匹配边$e\in M$至多与 2 条不同的$e^*\in M^*$冲突，因此可以将M^*中未被选入的边的权重摊派给M中的边，使得每条$e\in M$被分摊的总权重不超过其自身的 2 倍。于是有：$\sum_{e^*\in M^*}W^t(e^*)\leqslant 2\times\sum_{e\in M}W^t(e)$，因此，静态调度算法的近似比满足：$D(M)/D(M^*)\geqslant 1/2$。

证毕.

定理 6. 对于静态全局更新算法，1/2 的近似比是紧界限。

证明. 考虑如图 3 所示的 2 维超立方体图，图中节点 a~g 构成超立方体节点，每个超立方体节点均连接 2 个非超立方体节点。连接 2 个超立方体节点的边称为超立方体边，连接超立方体节点与非超立方体节点的边称为非超立方体边。假设所有非超立方体边的权重均为 1，所有超立方体边的权重均为$1+\varepsilon$，其中ε为任意小的正数。由于静态调度算法采用贪心策略，优先选择权重较高的边，因此将选择 4 条权

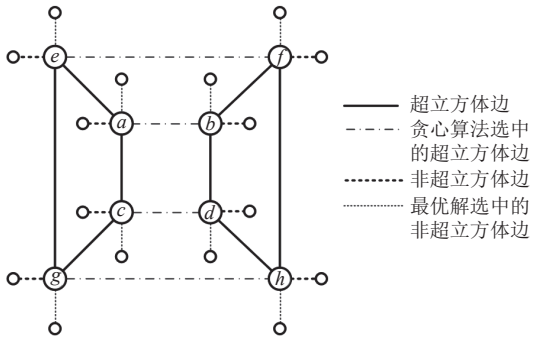


Fig. 3 2D hypercube graph

图 3 二维超立方体图

重为$1+\varepsilon$的超立方体边构成匹配，用图中较细的点划线表示。而最优解则包括 8 条权重为 1 的非超立方体边，用较细的虚线表示。因此，该实例中的近似比为：$\frac{4(1+\varepsilon)}{8}=\frac{4+4\varepsilon}{8}=\frac{1}{2}+\frac{\varepsilon}{2}$，当$\varepsilon\to 0$时，近似比趋近于 1/2，因此 1/2 是该算法的紧界限。

证毕.

4 仿真实验与结果分析

4.1 实验环境与测试数据集

本节通过仿真实验对 SYNC 与 4 种算法进行性能比较，包括实际吞吐率以及运行时间。所有实验均在配备 Intel® Core™ i5-1035G1 CPU(基准频率为 1.00 GHz，可睿频至 1.19 GHz)和 16.0 GB RAM 的计算机上运行。实验算法均使用 C++17 实现，并在同一编译环境下编译执行，以确保对比公平。采用以下真实生产环境数据集以及大型合成数据集：

1)真实生产环境数据集。本研究从 HPC 数据集[26]中获得了动态实例，这些数据集之前已被用于分析网络通信[27]。该数据集包含源节点-目标节点对，每个对都关联 1 个唯一的序列号。根据文献 [14] 中的处理方法，将 1 000 个序列号分组为一个更新批次，对于每个源-目标对(u,v)，统计该批次中出现的次数，并对其数据包大小求和，将其作为图中边(u,v)的新权重。处理后的实例包含 1 109 个时间戳，在单个时间戳内，网络最多包含 1 000 个有效节点和有效边，重复节点和边不计数。

2)大型合成数据集。本研究采用基于 Graph 500 RMAT 随机图模型[28]生成合成动态图数据集。该模型因其能够生成具有真实世界网络拓扑特性的随机图而被广泛应用于相关研究，例如 b-匹配的评估实验[29]。根据文献 [16,30] 的参数设置，本研究将 RMAT 生成器的初始矩阵参数设置为 0.25, 0.25, 0.25, 0.25，节点数$n=2^x$，其中$14\leqslant x\leqslant 18$。在这些合成动态图中，任意 2 节点间边的出现概率均等，且边权重遵循指数分布，取值范围为 [1, 500 000]。为模拟动态流量变化，实验共采用了 30 个连续的时间戳实例，代表了 30 s 内的流量输入演变情况。

对于 OCS 的设置，本研究参照 Helios[22]中使用的 Glimmerglass 公司生产的 64 端口 OCS 设备，并设定相应的光纤负载容量。同时，依据 Hanauer 等人[14]的研究，OCS 的数量选自集合 {2, 8, 32, 64}，以研究不同数量 OCS 对算法性能的影响。

4.2 性能评估

为全面对比本研究提出的算法的性能，实验选取以下 4 种具有代表性的算法进行对比：

1)Helios。该算法利用多个 OCS 设备来提升网络总吞吐率。其核心机制是贪心地为每个 OCS 选择具有最大流量的当前配置，时间复杂度为$O(km^tn^2)$[22]。

2)k-DM。该算法通过在每个 OCS 上独立应用 Blossom 算法[20]来获取k个最大边权重不相交匹配[16]，其理论近似比上限为 7/9。然而，该算法时间复杂度为$O(n^3)$，在大规模数据集上运行时间极长。在超过 10 000 个节点的 RMAT 数据集中，处理单个时间戳耗时 160 min 以上，会影响其他算法的对比结果展示，故本文仅在 HPC 数据集上加入该算法进行性能对比。

3)Dynamic。此算法基于 Hanauer 等人[14]提出的 dyn-greedy 算法，在匹配过程中引入了对光纤数量和容量的约束，并对每个时间戳的匹配变化量设限以提高系统稳定性。

4)Leaf-PQ。根据 Wang 等人[23]提出的 Leaf 算法，通过引入优先队列对贪心选择过程进行优化，旨在降低算法的时间复杂度，从而提升运行效率。

实验采用的性能指标包括吞吐率和运行时间。吞吐率是反映数据中心网络性能的核心指标；运行时间直接反映算法的计算时间开销和实时响应能力，是衡量其实用性的核心指标。为了获得可靠的实验结果，每个实验重复 5 次并取其平均值。图 4 和图 5 展示了在 OCS 的数量$k\in\{2,8,32,64\}$的配置下的实验结果。每张箱线图反映了运行时间的分布情况，菱形标记表示运行时间的几何平均值，星形标记表示实际吞吐率的算术平均值。本文使用几何平均值来汇总算法的运行时间，因为它能更稳健地处理数据中常见的偏态分布并能恰当地反映加速比。对于实际吞吐率，本文采用算术平均值，因为它能直观地表示算法在每个调度周期内平均完成的数据传输量。

1)真实生产环境数据集

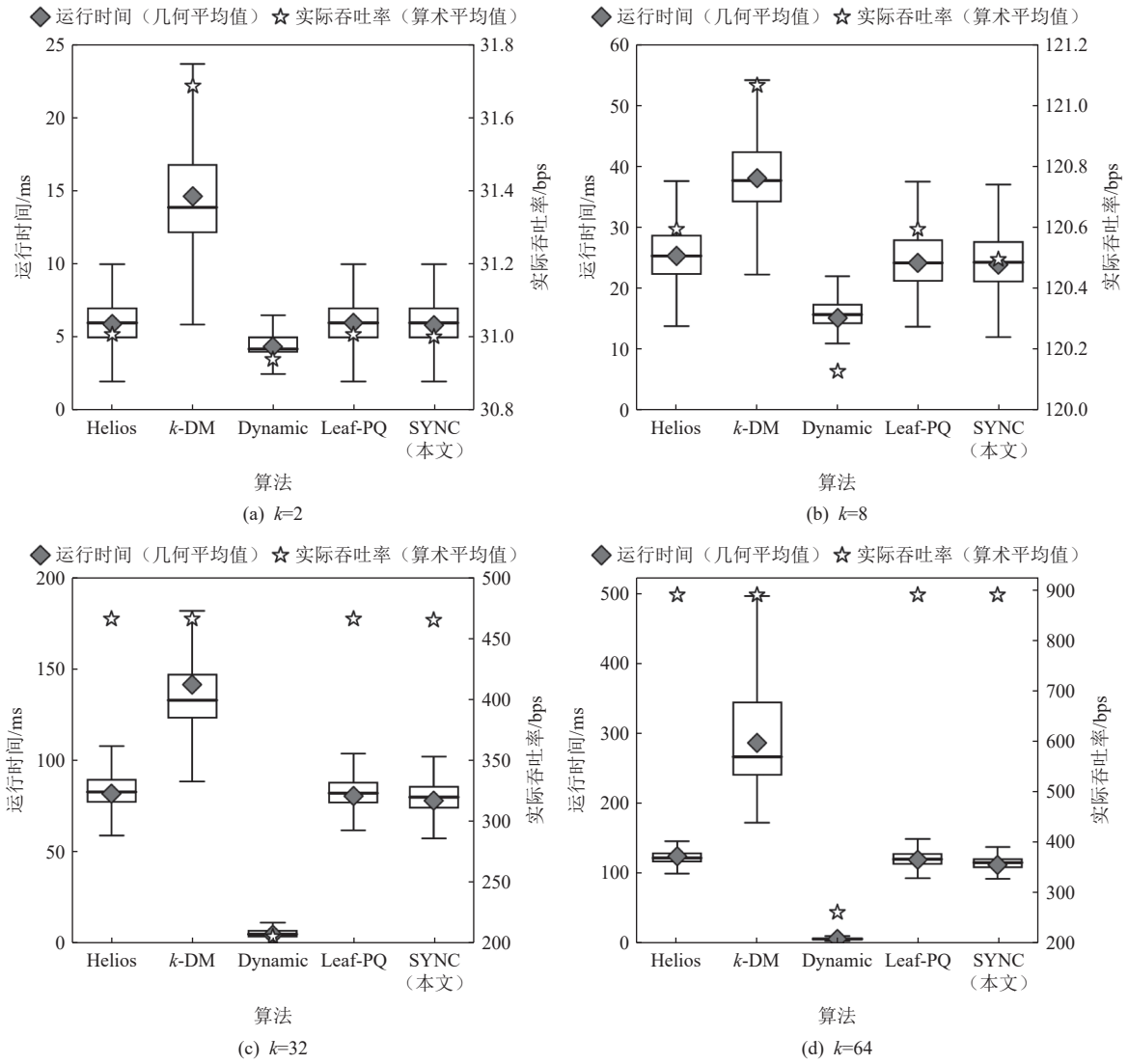


Fig. 4 Throughput and running time with different values of k on HPC dataset

图 4 HPC 数据集中不同k值下的吞吐率和运行时间

首先在真实世界实例上分析算法的性能。在实际吞吐率方面，如图 4 所示，不同算法的性能差异显著，且随着 OCS 数量的变化，其吞吐率结果也在动态变化。k-DM 算法凭借其 7/9 的近似比在所有情况下均获得了最高吞吐率。Leaf-PQ 与 Helios 算法凭借其静态贪心策略，获得了次高吞吐率，由于 HPC 数据集中边权重较低，故无法体现 Leaf-PQ 算法支持多台 OCS 协同为同一通信需求分配光纤资源的优势，2 个算法性能表现几乎完全相同，Leaf-PQ 的优势可在后文的合成实例中体现。与之相对，Dynamic 作为动态算法，其性能表现揭示了此类算法的局限性。在k值较小（如$k=2$）的资源受限场景下，如图 4(a)所示，其吞吐率与其他算法的差距尚不显著，较 Leaf-PQ 仅落后约 0.23%。然而，随着k值增大，Dynamic 算法的吞吐率与其他算法的性能差距被急剧拉大。这主要是因为动态算法仅关注局部情况且有稳定性阈值限制，容易产生“拓扑漂移”现象。而 SYNC 算法在吞吐率上始终表现出极强的竞争力，其结果与 Leaf-PQ 算法始终在保持 1% 以内的微小差距。如图 4(d)所示，在$k=64$时，SYNC 的平均吞吐率达到了 888.90 bps，与 Leaf-PQ 的 889.57 bps 和 k-DM 的 889.60 bps 相比，差距仅为 0.08%。

SYNC 算法在维持高吞吐率的同时，有效控制了运行时间。而k-DM 算法虽然保证极高吞吐率，但运行时间显著长于其他算法，在$k=64$时落后 SYNC 算法 60.98%。在网络规模较小时，如图 4(a)(b)所示，其他 3 种高吞吐率算法的运行时间差距较小，SYNC 仅展现出微弱优势。随着网络复杂度的增加，算法间的效率差异逐渐扩大。在$k=32$时，SYNC 的运行时间（78 ms）已明显优于 Leaf-PQ（81 ms）和 Helios（82 ms）。

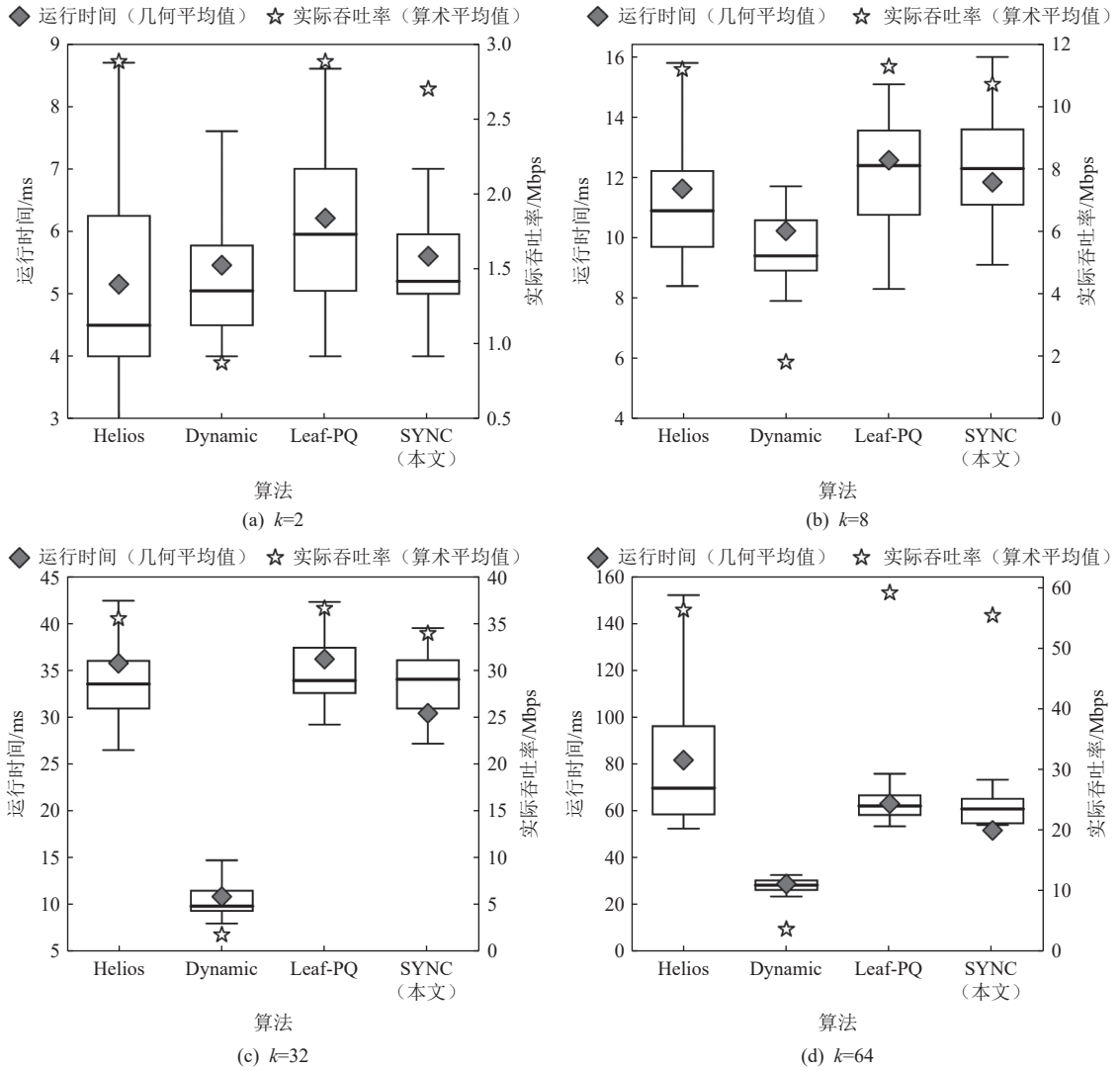


Fig. 5 Throughput and running time with different values of k on RMAT dataset

图 5 RMAT 数据集中不同k值下的吞吐率和运行时间

在$k=64$时，这一优势愈发显著，SYNC 的运行时间为 112 ms，与 Leaf-PQ(119 ms)和 Helios(124 ms)相比，分别取得了约 5.88% 和 9.68% 的性能提升。Dynamic 算法虽然始终保持最快的计算速度，但这是以牺牲巨大吞吐率为代价的。

2)大型合成数据集

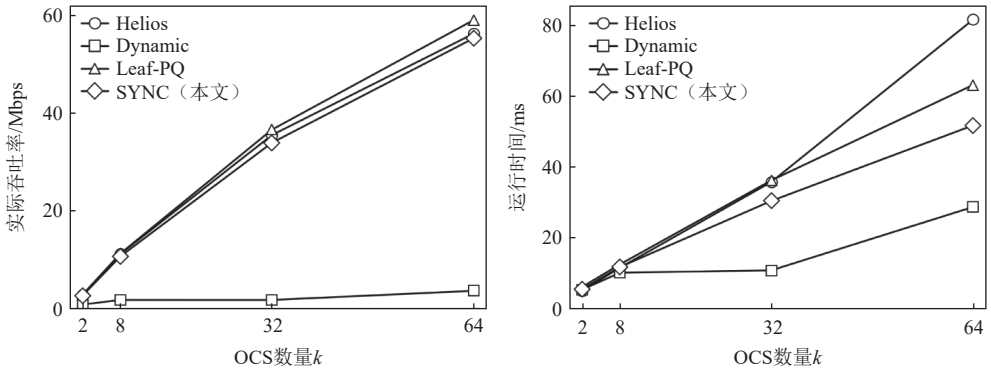
为进一步验证算法的性能表现，本研究还在大规模合成实例上进行了实验评估。

在实际吞吐率方面，Leaf-PQ 在所有k配置下均实现了最高的实际吞吐率。如图 5(a)(d)所示，其均值从$k=2$时的 2.88 Mbps 增长至$k=64$时的 59 Mbps，充分展示出其对资源的利用效率。Helios 在$k=2$时亦取得2.88 Mbps 的吞吐率，但在$k=8$，32，64时，其平均吞吐率分别为11.2 Mbps，35.4 Mbps，56.2 Mbps，均被 Leaf-PQ 超越。SYNC 的吞吐率则表现出稳定增长趋势，从2.7 Mbps 增至55.3 Mbps，且与 Helios 的差距随k的增大而缩小。在$k=32$(见图 5(c))时，其吞吐率为33.9 Mbps，较 Helios 的 35.4 Mbps 和 Leaf-PQ 的36.6 Mbps 分别少约 4.24% 和 7.38%；在$k=64$(见图 5(d))时，其吞吐率为 55.3 Mbps，较 Helios(56.2 Mbps)落后约 1.60%，较 Leaf-PQ(59 Mbps)落后约 6.27%。相比之下，Dynamic 算法因侧重快速响应，吞吐表现始终较低，维持在2～5 Mbps。

在运行时间方面，Dynamic 算法在k值较大时始终保持最低运行时间(5.5～29 ms)，且箱线图极为紧凑，展现出极佳的稳定性。在$k=2$时(见图 5(a))，Helios 的运行时间为5.2 ms，略优于 SYNC 的5.6 ms 与 Leaf-PQ 的6.2 ms，这是因为后二者的边权重插入与更新机制带来了额外的计算开销。随着 OCS 数量的增加，SYNC 的运行效率优势逐步显现并愈发突出。在

$k=8$(图 5(b))时，其运行时间与 Helios 基本持平，且显著优于 Leaf-PQ 的 12.6 ms；在 $k=32$(图 5(c))时，其运行时间为 30.4 ms，较 Helios 的 35.8 ms 和 Leaf-PQ 的 36.2 ms 分别快约 15.08% 和 16.02%；在 $k=64$(图 5(d))时，其运行时间为 51.7 ms，领先 Helios(81.5 ms)与 Leaf-PQ(63.1 ms)约 36.56% 和约 18.07%)。此外，SYNC 的运行时间箱线图在所有配置下的分布始终较为集中，表现出极高的稳定性，证明了该协同框架在大规模网络中的有效性。

为直观展示各算法吞吐率性能的可扩展性，图 6(a)展示了它们的平均实际吞吐率随 OCS 数量 k 变化的整体趋势。在 OCS 数量较少的场景下，由于网络资源受限，边权重远大于光纤容量的情况更为常见。此时，静态算法倾向于优先选择高权重边填充有限匹配槽位，因此 Leaf-PQ 与 Helios 可能做出相似的决策，从而取得相近的吞吐结果。导致 Leaf-PQ 和 SYNC 在此设置下的流量拆分机制难以充分发挥优势。然而，随着 OCS 数量的增加，资源约束逐步缓解，边权重拆分能力的重要性凸显。Leaf-PQ 依托优先队列机制和全局视角，在资源充足场景中仍能最优调度所有流量。SYNC 在静态调度阶段继承了此机制，配合动态部分的调节能力，实现了在吞吐率上的稳定增长与资源的高效利用。当 OCS 数量较多时，Helios 则因各 OCS 独立匹配、缺乏协同导致难以充分调度高权重边。相比之下，Leaf-PQ 和 SYNC 在处理大边权重时更具优势。



(a) 实际吞吐率（算术平均值） (b) 运行时间（几何平均值）

Fig. 6 Variation trends for throughput and running time of the four algorithms with different values of k on RMAT dataset

图 6 RMAT 数据集中不同k值下 4 个算法的吞吐率和运行时间的变化趋势

图 6(b)展示了各算法的运行时间随 OCS 数量 k 变化的整体趋势。当网络规模较小时，各高吞吐率算法的计算开销尚无显著差异。然而，随着 k 值增大，Helios 与 Leaf-PQ 的运行时间显著延长。与此形成鲜明对比的是，SYNC 的运行时间增长曲线表现出更优的缓和度，能更快速地响应网络需求。

综上，SYNC 凭借其静态-动态协同机制，在吞吐率与运行时间 2 方面均实现了良好平衡。在资源受限的极端场景下，其性能略受其机制复杂性的影响；而在中高规模网络下，SYNC 不仅具备较强的吞吐能力，还在运行效率和波动控制方面表现卓越，相较于其他 2 个对比算法，它以较小的吞吐率损失，获得了在大规模、高复杂度网络场景下显著的运行时间优势和更佳的可扩展性。

4.3 算法参数敏感性测试

SYNC 参数 *staticInterval* 的取值影响静态算法与动态算法的执行次数比例。为确定最优取值，基于 RMAT 数据集，将 OCS 数量 k 设置为 64，对 *staticInterval* 进行敏感性测试，实验结果如图 7 所示。当 *staticInterval* = 1时，系统仅执行静态算法，虽然吞吐率最高(约59 Mbps)，但其运行时间也最长(约 64 ms)，

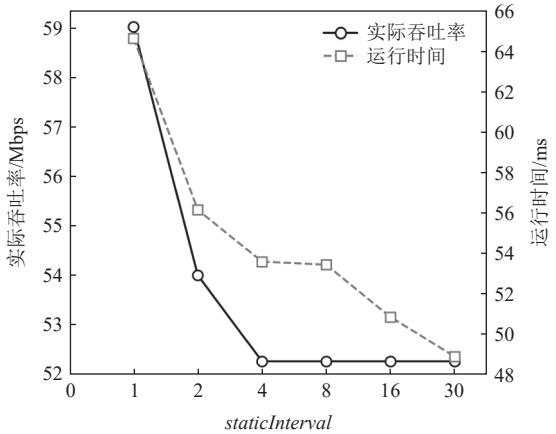


Fig. 7 Sensitivity analysis of parameter *staticInterval* on RMAT dataset ($k=64$)

图 7 在 RMAT 数据集上的参数*staticInterval*敏感性分析（$k=64$）

表明过度频繁的全局更新引入了显著时延开销。当*staticInterval*增至2, 运行时间显著优化至56 ms, 同时吞吐率仍保持在54 Mbps的较高水平。进一步增加*staticInterval*, 吞吐率会急剧下降并稳定在52.3 Mbps, 此后由动态算法决定性能上限, 而运行时间的优化幅度有限。综合考虑吞吐率与运行时间, *staticInterval* = 2为最优参数配置, 它在保持高吞吐率的同时, 有效降低了时延开销。

对动态局部调度算法参数α以及β的敏感性测试结果分别如图8和图9所示。对α的敏感性测试中, 为避免β对结果的影响, 其取值设置为β=6 000, 吞吐率与运行时间结果如图8所示。当$\alpha > 1$时, 吞吐率不会产生明显优化, 反而会导致运行时间增加。同时如图9所示, 当β=400时, 算法在2个指标上取得了较优的平衡; 当$\beta \geqslant 500$时, 其运行时间已经超过了

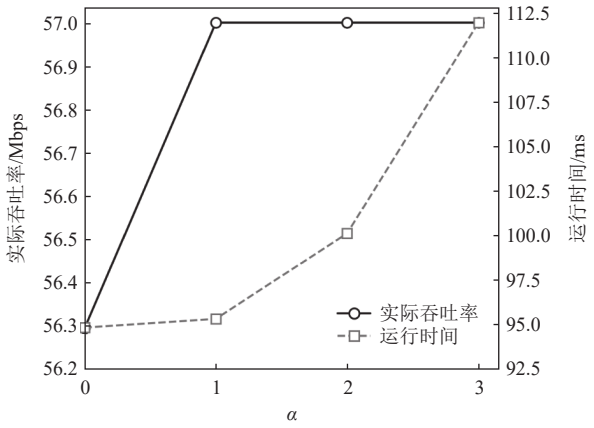


Fig. 8 Sensitivity analysis of parameter α on RMAT ($k = 64$, $\beta = 6000$)

图8 在RMAT上的参数α敏感性分析($k = 64$, β = 6 000)

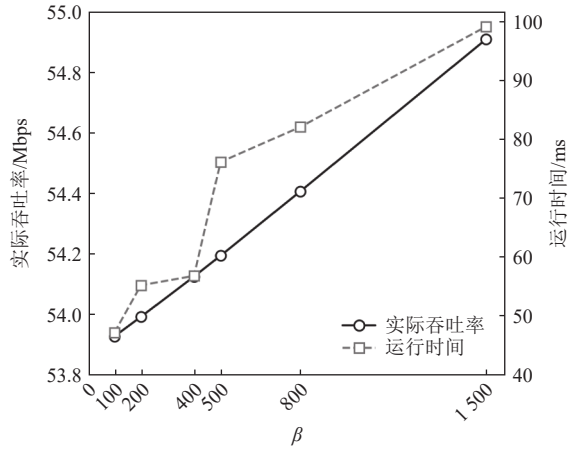


Fig. 9 Sensitivity analysis of parameter β on RMAT ($k = 64$, $\alpha = 1$)

图9 在RMAT上的参数β敏感性分析($k = 64$, $\alpha = 1$)

Leaf-PQ, 但吞吐率反而不如它。故后续实验中, α以及β取值分别为α=1和β=400。

5 结 论

本文为解决当前可重配置数据中心中重配置与流量调度算法在网络性能与运行时间之间难以权衡的问题, 首先对多OCS协同工作的场景进行了建模, 形式化为更符合实际的k-权重受限匹配问题, 并从理论上证明了该问题的NP困难性及其不可近似性。基于优先队列和贪心迭代设计了具有1/2近似比的静态全局更新算法, 以及时间复杂度较低的动态局部调度算法, 分别满足系统的全局优化与局部响应需求。为平衡这2种算法在运行效率与调度效果方面的优劣, 本文进一步提出了SYNC算法, 基于静态-动态协同策略, 通过结合时间周期与结构感知机制, 实现了调度模式的自适应切换。最后, 本文基于真实生产环境数据集、大型合成数据集进行了对比实验, 实验结果表明, 该算法在不同规模的光交换网络中均展现出良好的性能表现, 尤其是在中高负载场景下能够有效兼顾吞吐率与运行时间, 具有较强的实用价值和应用前景。

但是该算法仍存在进一步优化的空间, 未来的研究将重点探索动态调度算法的效率提升, 通过研究更高效的SwapIn/SwapOut候选边查找机制或设计新的低开销局部调整策略, 以进一步降低算法在高动态流量场景下的调度延迟。此外, 鉴于实验中观察到在OCS数量较少时, 简单的贪心策略因其计算开销较低而表现更佳, 未来可探索在SYNC算法内引入基于OCS数量的自适应算法选择机制, 例如, 在k值低于某一阈值时自动采用类似于Helios的简单贪心策略, 而在k值较大时则切换回当前更精密的静态-动态协同策略, 从而在更广泛的场景下实现最优的性能表现。

作者贡献声明: 赵佳欣提出了核心问题模型和初步算法设计, 进行了实验验证, 并完成了论文主要内容的撰写; 王竞舟改进了算法设计, 提出了实验方案; 孙玉娥对论文结构和实验设置提出了优化改进建议; 黄河对论文整体内容提出了指导意见。

参 考 文 献

[1] Wang Jingzhou, Zhao Gongming, Xu Hongli, et al. Trust: Real-time

request updating with elastic resource provisioning in clouds[C]//Proc of the 41st IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2022: 620−629

[2] Mellette W M, Snoeren A C, Porter G. P-FatTree: A multi-channel datacenter network topology[C]//Proc of the 15th ACM Workshop on Hot Topics in Networks. New York: ACM, 2016: 78−84

[3] Sultan M, Imbuido D, Patel K, et al. Designing knowledge plane to optimize leaf and spine data center[C]//Proc of the 13th IEEE Int Conf on Cloud Computing (CLOUD). Piscataway, NJ: IEEE, 2020: 13−15

[4] Schmidt S, Zerwas J, Kellerer W. Adfat: Adversarial flow arrival time generation for demand-oblivious data center networks[C]//Proc of the 19th Int Conf on Network and Service Management (CNSM). Piscataway, NJ: IEEE, 2023: 1−5

[5] Chen Jiajia, Cheng Weiqing. Analysis of web traffic based on HTTP protocol[C]//Proc of the 24th Int Conf on Software, Telecommunications and Computer Networks (SoftCOM). Piscataway, NJ: IEEE, 2016: 1−5

[6] Avin C, Schmid S. Revolutionizing datacenter networks via reconfigurable topologies[J]. Communications of the ACM, 2025, 68(6): 44−53

[7] Taubenblatt M, Maniotis P, Tantawi A. Optics enabled networks and architectures for data center cost and power efficiency[J]. Journal of Optical Communications and Networking, 2021, 14(1): A41−A49

[8] Ballani H, Costa P, Behrendt R, et al. Sirius: A flat datacenter network with nanosecond optical switching[C]//Proc of the ACM SIGCOMM 2020 Conf. New York: ACM, 2020: 782−797

[9] Wang Jingzhou, Zhao Gongming, Xu Hongli, et al. A robust service mapping scheme for multi-tenant clouds[J]. IEEE/ACM Transactions on Networking, 2021, 30(3): 1146−1161

[10] Sato K I. Optical switching will innovate intra data center networks[invited tutorial][J]. Journal of Optical Communications and Networking, 2023, 16(1): A1−A23

[11] Ye Xiaohui, Yin Yawei, Yoo S J B, et al. DOS: A scalable optical switch for datacenters[C]//Proc of the 6th ACM/IEEE Symp on Architectures for Networking and Communications Systems. New York: ACM, 2010: 1−12

[12] Xu Jing, Wang Zhan, Yuan Guojun, et al. Survey on fine-grained flow control for artificial intelligence data centers[J]. Journal of Computer Research and Development, 2025, 62(11): 2806−2825 (in Chinese)
(许晶, 王展, 元国军, 等. 面向智算中心的细粒度流量控制技术综述[J]. 计算机研究与发展, 2025, 62(11): 2806−2825)

[13] Wang Guohui, Andersen D G, Kaminsky M, et al. c-Through: Part-time optics in data centers[C]//Proc of the ACM SIGCOMM 2010 Conf. New York: ACM, 2010: 327−338

[14] Hanauer K, Henzinger M, Ost L, et al. Dynamic demand-aware link scheduling for reconfigurable datacenters[C]//Proc of the 42nd IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2023: 1−10

[15] Wang Guohui, Andersen D G, Kaminsky M, et al. Your data center is a router: The case for reconfigurable optical circuit switched paths[C]//Proc of the 8th ACM Workshop on Hot Topics in Networks (Hotnets). New York: ACM, 2009: 1−6

[16] Hanauer K, Henzinger M, Schmid S, et al. Fast and heavy disjoint weighted matchings for demand-aware datacenter topologies[C]//Proc of the 41st IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2022: 1649−1658

[17] Plander I, Stepanovsky M. Advanced three-dimensional MEMS photonic cross-connect switch for nonblocking all-optical networks[J]. Optical Switching and Networking, 2016, 22: 42−53

[18] Barker K J, Benner A, Hoare R, et al. On the feasibility of optical circuit switching for high performance computing systems[C]//Proc of the 18th ACM/IEEE Conf on Supercomputing (SC'05). Piscataway, NJ: IEEE, 2005: 16−16

[19] Mellette W M, McGuinness R, Roy A, et al. Rotornet: A scalable, low-complexity, optical datacenter network[C]//Proc of the ACM SIGCOMM 2017 Conf. New York: ACM, 2017: 267−280

[20] Edmonds J. Maximum matching and a polyhedron with 0, 1-vertices[J]. Journal of Research of the National Bureau of Standards Section B Mathematics and Mathematical Physics, 1965, 69B(1/2): 125−130

[21] Gabow H N. An efficient implementation of Edmonds' algorithm for maximum matching on graphs[J]. Journal of the ACM, 1976, 23(2): 221−234

[22] Farrington N, Porter G, Radhakrishnan S, et al. Helios: A hybrid electrical/optical switch architecture for modular data centers[C]//Proc of the ACM SIGCOMM 2010 Conf. New York: ACM, 2010: 339−350

[23] Wang Jingzhou, Zhao Gongming, Xu Hongli, et al. Leaf: Improving QoS for reconfigurable datacenters with multiple optical circuit switches[C]//Proc of the 32nd IEEE/ACM Int Symp on Quality of Service (IWQoS). Piscataway, NJ: IEEE, 2024: 1−10

[24] Berge C, Fournier J C. A short proof for a generalization of Vizing's theorem[J]. Journal of Graph Theory, 1991, 15(3): 333−336

[25] Holyer I. The NP-completeness of edge-coloring[J]. SIAM Journal on Computing, 1981, 10(4): 718−720

[26] Avin C, Ghobadi M, Griner C, et al. On the complexity of traffic traces and implications[J]. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 2020, 4(1): 1−29

[27] Bienkowski M, Fuchssteiner D, Marcinkowski J, et al. Online dynamic b-matching: With applications to reconfigurable datacenter networks[J]. ACM SIGMETRICS Performance Evaluation Review, 2021, 48(3): 99−108

[28] Chakrabarti D, Zhan Yiping, Faloutsos C. R-MAT: A recursive model for graph mining[C]//Proc of the 4th SIAM Int Conf on Data Mining. Philadelphia, PA: SIAM, 2004: 442−446

[29] Khan A, Pothen A, Patwary M M A, et al. Efficient approximation algorithms for weighted b-matching[J]. SIAM Journal on Scientific Computing, 2016, 38(5): S593−S619

[30] Khan A, Pothen A. A new 3/2-approximation algorithm for the b-edge cover problem[C]//Proc of the 7th SIAM Workshop on Combinatorial Scientific Computing. Philadelphia, PA: SIAM, 2016: 52−61



Zhao Jiaxin, born in 2004. Master candidate. Her main research interests include reconfigurable network and data center.

赵佳欣, 2004 年生。硕士研究生。主要研究方向为可重配置网络、数据中心。



Wang Jingzhou, born in 1997. PhD, associate professor. His main research interest includes data center networks.

王竞舟, 1997 年生。博士，副教授。主要研究方向为数据中心网络。



Sun Yu'e, born in 1983. PhD, professor. Her main research interests include crowdsourcing and traffic measurement.

孙玉娥, 1983 年生。博士，教授。主要研究方向为众包、流量测量。



Huang He, born in 1983. PhD, professor. His main research interests include traffic measurement and compact data structure.

黄　河, 1983 年生。博士，教授。主要研究方向为流量测量、紧凑数据结构。