面向移动机器人端边协同推理的混合数据流调度算法研究

丁 男[1,2],方玺淇[1],郝云涛[1],胡创业[1],许 力[3]

(1. 大连理工大学计算机科学与技术学院,辽宁大连 116024;
2. 大连理工大学工业装备智能控制与优化教育部重点实验室,辽宁大连 116024;
3. 北京字节跳动科技有限公司,北京 100101)

摘 要: 基于分区的深度神经网络(Deep Neural Network,DNN)端边协同推理技术通过将模型拆分并分别在移动机器人终端和边缘服务器上部署,能够有效缓解端设备的资源受限以及现有模型轻量级化技术导致的推理精度降低等问题. 然而,该技术也为机器人操作系统(Robot Operating System2,ROS2)的通信调度提出了新的挑战:现有的通信策略难以在保障协同推理关键数据流有效传输的同时,兼顾其他应用数据流的传输需求. 针对这一问题,本研究提出了机器人操作系统中面向移动机器人深度神经网络端边协同推理的混合数据流动态调度算法(Hybrid Data Flow Dynamic Scheduling Algorithm for Mobile Robot Deep Neural Network Edge-End Collaborative Inference in the Robot Operating System2,DRECHS). 首先,基于端边协同推理的机理分析,定义了深度神经网络中间数据的最大允许传输时间边界条件,为传输优化提供了理论基础. 结合边界条件,设计了一种基于混杂切换系统理论的调度模型,将流调度过程建模为包含优先级优先子系统和时间优先子系统的动态切换模型. 在此基础上,提出了具体的混合数据流调度算法. 该算法集成在机器人操作系统的数据分发服务(Data Distribution Service,DDS)流控制器中,能够依据计算出的队列状态指标动态生成输出队列,实现对底层数据传输顺序的细粒度控制,从而在满足推理任务数据传输要求的基础上,实现对不同优先级数据流的差异化服务质量(Quality of Service,QoS)优化,有效平衡了系统的整体传输性能. 针对所采用的动态分区方法,设计不同带宽条件下的仿真实验,对比分析了所提算法与系统内置调度算法等在传输延迟和丢包率方面的性能差异. 实验结果表明,本研究提出的调度算法通过混杂切换系统模型和动态调度策略,在满足高优先级数据传输需求的同时,成功实现了对不同优先级数据流的差异化服务质量优化. 此外,本研究提出了相应的部署方案,并在真实设备上部署了该调度算法及深度神经网络端边协同推理框架,完成了系统验证. 该部署方案为本研究所提算法及框架在真实场景中的部署提供了参考.

关键词: 深度神经网络;端边协同;机器人操作系统;混合数据流调度;混杂切换系统理论
基金项目: 国家自然科学基金(No.62262066,No.62473071)
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2025)12-4575-17
电子学报 URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20250674

Research on Hybrid Data Flow Scheduling Algorithm for Mobile Robot Edge-End Collaborative Inference

DING Nan[1,2], FANG Xi-qi[1], HAO Yun-tao[1], HU Chuang-ye[1], XU Li[3]

(1. *School of Computer Science and Technology, Dalian University of Technology, Dalian, Liaoning* 116024, *China*;
2. *Key Laboratory of Intelligent Control and Optimization for Industrial Equipment, Dalian University of Technology, Dalian, Liaoning* 116024, *China*; 3. *ByteDance Inc., Beijing* 100101, *China*)

Abstract: Partition-based edge-end collaborative inference technology for deep neural networks (DNN), by splitting models and deploying them on mobile robot terminals and edge servers respectively, can effectively alleviate resource constraints on terminal devices and address the issue of reduced inference accuracy caused by existing model lightweighting techniques. However, this technology also poses new challenges for communication scheduling in the robot operating system2 (ROS2): existing communication strategies struggle to ensure the effective transmission of critical collaborative infer-

收稿日期:2025-08-04;录用日期:2025-12-12;责任编辑:覃怀银

ence data flows while simultaneously accommodating the transmission needs of other application data flows. To address this problem, this study proposes a hybrid data flow dynamic scheduling algorithm for mobile robot deep neural network edge-end collaborative inference in the robot operating system2 (DRECHS). First, based on the mechanism analysis of edge-end collaborative inference, we define the maximum allowable transmission time boundaries for DNN intermediate data to provide a theoretical basis for transmission optimization. Combining these boundary conditions, we design a scheduling model based on hybrid switching system theory, modeling the flow scheduling process as a dynamic switching model containing a priority-first subsystem and a time-first subsystem. On this basis, the specific hybrid data flow scheduling algorithm is proposed. Integrated into the data distribution service (DDS) flow controller of the robot operating system2, this algorithm is capable of dynamically generating output queues based on calculated queue status metrics, realizing fine-grained control of the underlying data transmission order. Thus, while meeting the transmission requirements of inference tasks, it achieves differentiated quality of service (QoS) optimization for data flows with different priorities, effectively balancing overall system transmission performance. Targeting the adopted dynamic partitioning method, we design simulation experiments under different bandwidth conditions to compare and analyze the performance differences between the proposed algorithm and the system´s built-in scheduling algorithms and others in terms of transmission delay and packet loss rate. Experimental results show that the proposed scheduling algorithm, through the hybrid switching system model and dynamic scheduling strategy, successfully achieves differentiated quality of service optimization for data flows with different priorities while meeting high-priority data transmission requirements. Furthermore, this study proposes a corresponding deployment scheme, and deploys the scheduling algorithm and the deep neural network edge-end collaborative inference framework on real devices, completing system verification. This deployment scheme provides a reference for the deployment of the proposed algorithm and framework in real-world scenarios.

Key words: deep neural networks; edge-end collaboration; robot operating system2; mixed data flow; hybrid switching systems

Foundation Item(s): National Natural Science Foundation of China (No.62262066, No.62473071)

1 引言

随着5 G和人工智能技术的快速发展,端边协同框架逐渐成为机器人系统应用的主要发展趋势,例如智能网联汽车[1]、智能制造机器人等. 该框架通过将计算任务分布在端设备和边缘设备,能够显著提高系统的资源利用效率和任务完成效率,相比于将计算任务迁移至云端[2],更好地解决了可用性、延迟、带宽及安全性等挑战. 具体来说,机器人可以借助边缘设备的高效计算能力,提升自主决策和协同工作的效率[3]. 其中,深度神经网络(Deep Neural Network,DNN)端边协同推理是关键技术. 如图1所示,通过在移动端和边缘端的服务器之间动态划分深度神经网络的推理任务,既能降低端设备的计算负载,又能减少数据传输时延,有效满足了机器人系统对实时性和计算效率的需求.

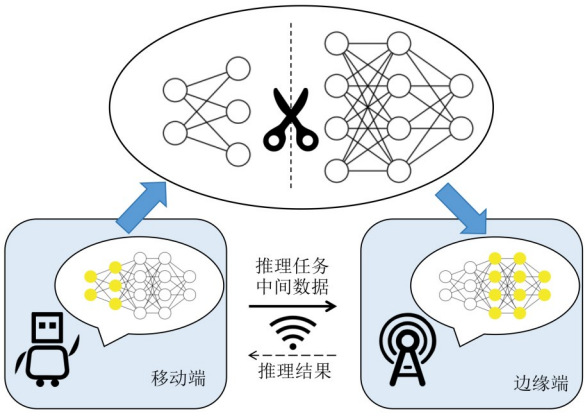


图1 端边协同示意图

为了支持复杂的端边协同任务,机器人操作系统(Robot Operating System2,ROS2)作为一种现代机器人开发框架,凭借其分布式架构和实时通信能力[4],为机器人系统端边协同推理提供了理想的实现平台. ROS2采用数据分发服务(Data Distribution Service,DDS)作为通信中间件[5],通过服务质量(Quality of Service,QoS)策略[6]和流控制器机制实现数据流的可靠传输和速率控制,并在流控制器中内置了多种调度算法. 图2展示了系统的分层架构设计,包括应用层的节点模块、中间件层的客户端库和核心服务组件,以及底层的多平台系统支持. 这种架构通过中间件层提供统一的接口,实现了跨平台的数据传输服务和安全保障机制,为端边协同应用提供了稳定高效的通信基础.

然而,在实际应用中,DNN端边协同推理任务往往涉及多种类型的混合数据流,Saboia等人[7]研究机器人通信时,将数据传输类型主要分为3类:Mission-critical data(任务关键型数据)、Time-sensitive data(时间敏感数据)和Key data(普通关键数据). 这些数据具有不同的传输需求,ROS2中DDS现有的QoS策略以及内置调度策略难以有效处理这种混合数据流的差异化传输需

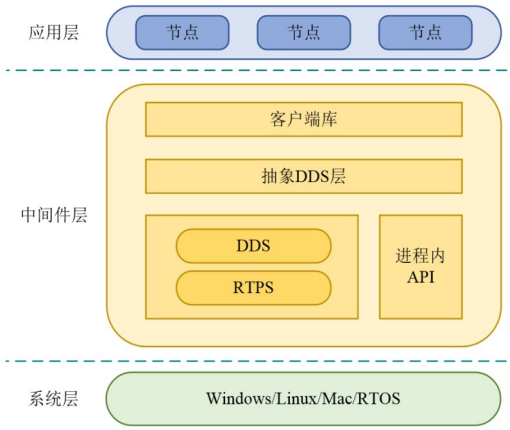


图2　ROS2架构图

求,在保证推理任务数据传输性能的同时兼顾其他类型数据的传输特性,已成为亟待解决的关键问题.

本研究从DNN端边协同的实际应用需求出发,提出了一套系统化的混合数据流调度解决方案. 首先依据端边协同推理通用基本耗时构成,定义了DNN中间传输数据的最大允许传输时间边界条件,为混合数据流传输优化提供了理论基础,将上层应用的需求映射到底层通信参数. 然后,结合边界条件定义,通过将消息数据的调度过程建模为混杂切换系统[8],设计了ROS2中面向移动机器人深度神经网络端边协同推理的混合数据流动态调度算法(Hybrid Data Flow Dynamic Scheduling Algorithm for Mobile Robot Deep Neural Network Edge-End Collaborative Inference in the Robot Operating System2,DRECHS). 该算法通过动态调度策略,实现了对不同类型数据传输需求的优化. 最后,将DRECHS算法集成到ROS2中DDS的流控制器当中,在DNN端边协同过程传输的多条数据流中,可根据实际需求对需要差异化服务质量的数据流进行优化调度. 结合实验具体选用的动态分区算法,通过仿真实验,详细分析了DRECHS算法与ROS2内置调度算法在传输延迟和丢包率方面的性能差异,并在真实设备上完成了对DNN端边协同框架和调度算法的部署,验证了调度方法在真实设备上运行的可行性和稳定性.

本文的主要研究工作如下.

(1)本研究针对移动机器人DNN端边协同场景中推理任务数据流与其他数据流的并发调度冲突问题,结合ROS2中消息传输架构特点,提出了一种基于混杂切换系统的调度模型与DRECHS算法,实现了对混合数据流的按需优化调度.

(2)定义了DNN推理数据的“最大允许传输时间”作为边界条件,为调度决策提供了量化依据. 该边界为DRECHS调度算法针对“优先级优先”与“时间优先”两个子系统在设置切换规则时提供理论参考.

(3)针对ROS2架构DDS流控制器现存的调度算法对时间敏感以及优先级优先等混合并发的数据流调度的不足,结合DRECHS调度算法和ROS2的特点进行集成,以提供差异化的数据流服务质量保障. 最终,通过动态分区方法的仿真实验与真实设备部署,对算法的有效性进行了验证.

2　相关工作

传统机器人系统通过将计算任务迁移至云端显著提升了系统的服务能力,但同时也带来了系统可用性、通信延迟、网络带宽占用以及数据安全性等多重挑战. 为应对这些问题,边缘计算与雾计算逐渐发展成为主流解决方案[3]. 有关边缘计算的研究中,陈阳等人[9]针对多无人机协同陆地设施辅助边缘计算系统,提出了一种多子群驱动的均衡优化算法,通过混合编码策略联合优化无人机部署位置与用户卸载决策,有效解决了系统能耗最小化问题;许小龙等人[10]针对车联网边缘计算中交通流时空分布不均导致的边缘服务器负载失衡问题,提出了切比雪夫图加权网络进行流量预测,并结合基于深度强化学习的二元任务卸载算法,通过两阶段决策优化实现了系统总计算速率的最大化与服务响应速度的提升. 在机器人端边协同系统研究中,计算任务卸载[11](特别是深度目标识别和目标检测等计算密集型任务)和网络通信优化[12]成为研究重点. 随着人工智能技术的快速发展,现代移动设备,例如智能手机、自动驾驶汽车、移动机器人等,日益依赖深度神经网络来完成图像分类、语音识别等复杂推理任务. 然而,在设备端直接运行完整的DNN模型往往会导致计算资源的严重消耗. 虽然传统方法通过将计算任务完全卸载到云端或边缘服务器来降低设备端的计算负担,但网络状况的波动性和服务器负载的不稳定性可能引发显著的延迟问题[3].

为解决这一挑战,研究者提出了基于拆分计算(Split Computing, SC)的方法[13],该方法通过将DNN模型划分为头部模型和尾部模型,并分别部署在移动设备和边缘服务器上执行,既有效降低了带宽占用,又显著减少了设备能耗. 在该领域的代表性工作中,Kang等人[14]首次提出了SC方法Neurosurgeon,通过迭代搜索最佳分割层来最小化端到端推理延迟或能耗;Hu等人[15]将图论中的最大流最小割算法应用于分析DAG拓扑结构,有效解决了具有特定拓扑结构的DNN模型的划分问题;Pagliari等人[16]针对简单递归神经网络的协同推理场景,提出了基于输入数据特征的动态推理设备选择方案,实现了总体延迟和终端能耗的优化. 然而,在当前DNN边端融合机器人系统中,信息呈现出多种类型数据流的混合特性,如何确保这种混合数据流

通信的可靠性成为亟待解决的关键问题. 在机器人端边协同计算研究领域,针对实际应用场景中如何实现对混合数据流差异化调度策略的研究仍显不足,尤其缺乏对不同数据类型特性的精细化处理机制.

在多机器人系统中,ROS2因其支持DDS和灵活的QoS参数配置成为数据传输协议的优选方案. 与ROS相比,ROS2在多机器人通信场景下表现更加出色,能够有效应对复杂网络环境. 然而,尽管ROS2通过QoS策略优化了数据传输,其在处理多样性混合消息时,消息数据的优先级调度以及消息数据的实时性和可靠性保障仍然存在不足. 近年来,针对ROS2通信性能优化的研究成为热点[17],提出了诸如调整QoS配置、引入优先级驱动调度算法、采用部分序列化技术以及开发自适应序列化算法等多种策略,旨在提高数据传输效率、减少延迟并增强系统的整体性能.

针对ROS2通信性能的研究主要集中在传输速度、延迟和吞吐量等方面. 例如,Maruyama等人[18]和Kronauer等人[19]对ROS2的通信性能进行了系统性评估. 一些研究还通过高效的数据压缩算法、优化的序列化方法以及更快的网络协议进一步提升通信性能. 例如,Jiang等人[20]提出了自适应双层序列化算法,通过将部分处理转移至编程接口层来优化ROS2中的消息传递;Wang等人[21]改进了ROS2中的部分序列化方法,将消息分为通过套接字传输和共享内存传输的两部分,避免了共享内存部分的复制与序列化操作. 实验表明,其延迟性能相比ROS和ROS2,IPC有所提升. Luo等人[22]在研究中发现ROS2多订阅者环境下存在优先级反转问题,即高优先级订阅者可能与低优先级订阅者一样晚收到消息,并提出了基于优先级的策略和数据重用政策来解决这一问题. Teper等人[23]分析了多执行器ROS2系统中端到端延迟问题,提出了上界分析方法和优化策略,成功降低了端到端延迟上界和最大测量延迟. Castillo-Sánchez等人[24]评估了ROS2在机器人集群无线通信中的性能,发现默认设置下使延迟显著增加的节点数量,因此建议小型集群使用单播,大型集群启用多播并配置高比特率编码,以优化通信性能和减少带宽消耗. Arafat等人[25]通过修改ROS2原生的事件执行器,将其事件队列替换为优先级队列,使其兼容经典的非抢占式调度理论,显著降低了端到端延迟和作业丢弃率. Teper等人[26]为解决ROS2共享资源下的优先级反转问题,设计了基于优先级队列的动态多线程执行器,提升了系统性能.

此外,传统网络、数据中心网络、工业物联网和车联网中的通信优化方法为机器人网络优化提供了重要参考. 例如,文献[27]提出了面向多租户数据中心的关键流优先调度策略[27],通过多级反馈队列机制区分延迟敏感流和吞吐需求流,实现了数据流的平衡传输,显著降低了数据流的传输时间;同时,针对工业物联网对低延迟和高可靠性的需求,文献[28]基于混杂系统理论提出了一种时间敏感网络资源调度算法[28]. 该算法综合考虑数据包的优先级、传输时间和最大传输延迟等因素,通过仿真实验验证了其在满足高优先级数据传输需求的同时,能够有效控制待发送队列的平均传输时间,并显著降低系统丢包率.

基于上述研究的启发,本文将混杂切换系统理论应用于机器人网络通信领域,重点研究以ROS2通信中间件为基础,机器人DNN端边协同框架下的混合数据流的传输优化问题,旨在提升关键数据的传输可靠性和实时性.

3 调度框架、参数和边界条件分析

3.1 调度框架

本文研究的调度框架如图3所示,在机器人端边协同系统中,存在多种数据流,其中需要差异化服务质量的数据流通过ROS2的DDS中的FlowController模块进行处理. FlowController是DDS在RTPS层提供的流控制器机制,通过独立的异步发送线程,实现对使用相同流控制策略的DataWriter数据流发送速率和顺序的控制. 本研究聚焦于3种需要差异化服务质量的数据流在流控制器中的调度优化,即DNN推理任务数据、时间敏感数据和普通关键数据.

在端边协同框架下,ROS2移动端执行DNN前段推理并发布结果,边缘服务器接收数据完成后段推理. 具体而言,移动端的ROS2Publisher整合并发布这3种不同优先级的数据,当完成DNN前段推理后,将推理中间结果与其他类型数据一起作为混合数据流发布. 每个Publisher维护与数据优先级对应的发布队列,这些数据随后被整合到DDS中对应优先级的DataWriter队列.

本研究提出的DRECHS算法集成在FlowController中,负责对3种优先级的DataWriter队列数据进行统一调度,生成最终输出队列. 这些经过调度的数据被发送到边缘服务器对应优先级的DataReader队列,再由DataReader整合到相应的Subscriber队列. 最终,边缘服务器接收并处理这些经过差异化调度的混合数据流,完成DNN后段推理并输出分类结果.

3.2 调度参数

3.2.1 消息数据优先级

第1节已经简单叙述过,在机器人通信研究中,Saboia等人[7]将数据传输类型主要分为3类. 其中,任务关键型数据包含机器人DNN端边协同推理任务的中间传输数据等与具体任务相关的数据,需要优先保障传输可靠性,同时关注传输延迟的控制;时间敏感数据主

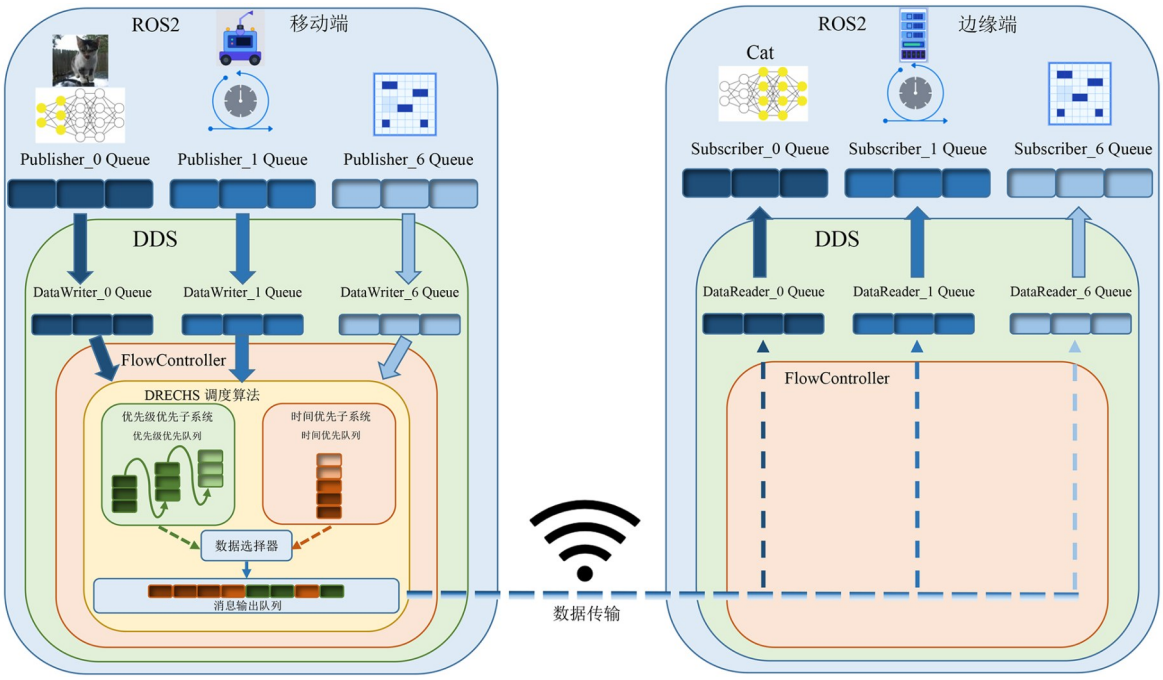


图3　调度框架

要包含机器人的位置信息等,要求较高的数据时效性和可靠性;普通关键消息数据包含增量地图等信息,虽无严格的时间限制,但需要保证定期传输. 本研究在DDS的流控制器中针对这3种需要差异化服务质量的数据类型进行调度优化.

在优先级机制设计方面,参考IEEE802.1p协议的定义,其协议头包含一个三位优先级字段,用于定义0~7共8个优先级等级. 其中,优先级7(最高)用于关键性网络流量,优先级6和5用于延迟敏感应用程序,优先级4到1用于受控负载应用程序,优先级0为默认值. 基于此协议标准并结合机器人通信数据的特点,本研究对ROS2中不同话题的通信数据及其数据写入者(Datawriter)进行优先级划分,并在负责封装传输消息的CacheChange_t类中增加priority属性字段,实现对消息数据的优先级标记.

具体的优先级定义如下:p_i表示消息数据的优先级,其中下标i标记了该类型消息数据优先级的数值,并且$i\in\{0,1,2,3,4,5,6,7\}$,i值越小表示优先级越高,这是因为通过对ROS2-DDS源代码中优先级机制的深入分析,发现其采用的是"数值递减表示优先级递增"的映射方式,为确保所提调度模型与DDS原生优先级机制的衔接,本研究遵循了该设定标准. 特别地,使用p_{mission}表示任务关键型消息数据的优先级,p_{time}表示时间敏感消息数据的优先级. 本研究假设上述3种数据类型以外的数据队列为空. 详细的数据优先级等级划分如表1所示.

表1　消息数据分类

数据类型	优先级等级	对应优先级
任务关键型	高	0
时间敏感型	中	1
普通关键型	低	6

3.2.2　消息数据优先级权重

消息数据优先级权重用于量化不同优先级消息数据的重要程度,定义为

$$\text{pw}_i=\frac{N-p_i}{N} \tag{1}$$

其中,N表示消息数据优先级的总数量(基于IEEE802.1p协议标准定义的优先级等级,$N=8$).

3.2.3　平均已等待时间

平均已等待时间用于表征当前待发送队列中消息数据的整体延迟情况,定义为

$$\bar{t}=\frac{\sum_{c=1}^{n}\left(1+\text{pw}_i\right)t_w}{n} \tag{2}$$

其中,t_w表示单个消息数据的已等待时间;n表示待发送队列中消息数据的总数.

$$t_w=t_{\text{now}}-t_{\text{in}} \tag{3}$$

其中,t_{now}表示当前系统时间;t_{in}表示消息数据进入队列的时间.

根据上述定义,在计算$\bar{t}$时,高优先级消息数据的等待时间具有较大权重. $\bar{t}$的值越大,表示当前队列中接近最大允许传输时间的消息数据越多,反映了队列中数据的传输紧迫程度.

3.2.4 加权传输完成时间

根据上述定义，在计算$\bar{t}$时，高优先级消息数据的等待时间具有较大权重．$\bar{t}$的值越大，表示当前队列中接近最大允许传输时间的消息数据越多，反映了队列中数据的传输紧迫程度．

加权传输完成时间用于量化队列中单个消息的加权传输延迟，定义为

$$t_{\mathrm{fw}}=r_f(p_i)\cdot t_f \tag{4}$$

其中，$r_f(p_i)$表示优先级为p_i的消息数据的传输完成时间权重系数，该系数反映了数据流的时间敏感度，系数越小，表示该数据流对时间延迟的敏感度越高．具体定义如下．

当$p_{\mathrm{time}}<p_i\leqslant p_0$时：

$$r_f(p_i)=r_f(p_{\mathrm{time}})+(i+1)\frac{r_f(p_{N-1})-r_f(p_{\mathrm{time}})}{N-1} \tag{5}$$

当$p_{N-1}<p_i<p_{\mathrm{time}}$时：

$$r_f(p_i)=r_f(p_{N-1})-(N-1-i)\frac{r_f(p_{N-1})-r_f(p_{\mathrm{time}})}{N-1} \tag{6}$$

其中，$r_f(p_{\mathrm{time}})$和$r_f(p_{N-1})$分别表示加权系数的最小值和最大值．t_f表示单个数据到达接收节点的理想截止时间，定义为

$$t_f=\mathrm{transmax}(p_i)+t_{\mathrm{in}} \tag{7}$$

其中，$\mathrm{transmax}(p_i)$为不同类型消息数据的最大允许传输时间(详见3.3节)．

加权系数的大小直接反映了消息数据对传输延迟的敏感程度：系数值越小，表明该消息对延迟越敏感，需要为其提供更快的传输服务；系数值越大，则表明该消息可以容忍较长的传输延迟．数据在时间子系统进行调度时，允许加权系数大的数据流适当让位于加权系数小的数据流，从而保证时间敏感度高的数据的传输质量．

3.3 边界条件分析

边界条件定义了不同类型消息数据的最大允许传输时间，对于优先级为p_i的消息数据，其最大允许传输时间反映了系统所允许的理想传输延迟．本研究重点针对任务关键型、时间敏感型和普通关键型这3类数据特征，分别制定了相应的最大允许传输时间定义．

3.3.1 普通关键消息

对于普通关键消息，其最大允许传输时间定义为

$$\mathrm{transmax}(p_i)=T(p_i) \tag{8}$$

其中，$T(p_i)$表示优先级为p_i的消息数据的发布周期．

3.3.2 任务关键型消息

在DNN端边协同推理框架耗时的评估中，框架通用基本整体耗时通常划分为3个部分：

$$t_{\mathrm{sum}}=t_m+t_t+t_e \tag{9}$$

其中，t_{sum}表示框架整体耗时；t_m表示移动端推理时间；t_t表示数据传输时间；t_e表示边缘端推理时间．

对于包含DNN推理任务的任务关键型数据，由于移动端必须先完成前段推理才能发布中间数据，这部分推理时间t_m会占用数据传输的可用时间．因此，其最大允许传输时间定义为

$$\mathrm{transmax}(p_{\mathrm{mission}})=T(p_{\mathrm{mission}})-t_m \tag{10}$$

其中，$T(p_{\mathrm{mission}})$表示任务关键型数据的发布周期．

该定义表明，在每个发布周期内，任务关键型数据的最大可用传输时间等于周期总时长减去移动端推理时间．特别地，当消息不包含DNN推理任务时，$t_m=0$，此时可利用整个发布周期进行传输，即$\mathrm{transmax}(p_{\mathrm{mission}})=T(p_{\mathrm{mission}})$.

3.3.3 时间敏感型消息

时间敏感型消息对延迟具有较高的约束要求．为确保系统的实时性能，其最大允许传输时间定义为其自身发布周期与所有优先级数据最大允许传输时间中的最小值：

$$\mathrm{transmax}(p_{\mathrm{time}})=\min\left(T(p_{\mathrm{time}}),\min_{i=0}^{N-1}\left(\mathrm{transmax}(p_i)\right)\right) \tag{11}$$

其中，$T(p_{\mathrm{time}})$表示时间敏感型数据的发布周期．这一定义保证了时间敏感型消息具有较高的时间约束．

4 混杂切换模型与调度算法

4.1 混杂切换模型

混杂系统是一类同时具有连续动态和离散事件动态特性的动态系统．这类系统的状态空间由连续状态变量和离散状态变量构成，其本质特征是系统行为同时包含连续时间演化与离散事件驱动的状态转换．切换系统作为混杂系统的一个重要子类，由多个具有不同动态特性的子系统组成．当系统状态满足预设的切换条件时，系统将按照特定的切换规则从一个子系统跃迁至另一个子系统．切换系统因其模型结构清晰、系统逻辑严谨且具有良好的通用性，已在控制系统、机器人系统、通信系统等诸多领域得到了广泛应用，展现出显著的理论价值和实践意义．

针对机器人网络中不同类型话题消息数据具有不同优先级等级和时延要求的特点，本文构建了一个包含优先级子系统和时间子系统的混杂切换模型来描述消息的调度过程，如图4所示．在该模型中，系统先将队列中的数据依据优先级和时间两个维度，分别映射到两个独立的子系统当中，随后，根据队列中消息数据的情况，在两个子系统之间进行动态切换，选择对应子系统的消息数据，从而实现对消息数据的调度．

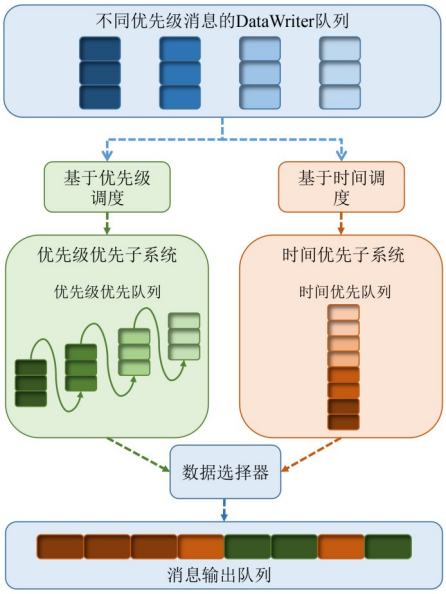


图4　调度模型示意图

在优先级优先子系统中,按照消息发布者的优先级从高到低对其队列进行排序,消息数据根据优先级p_i从高到低排列,若优先级相同,则按照消息数据的已等待时间t_w从大到小排列.

在时间优先子系统中,对所有优先级发布者的消息数据进行重新整合,将待发送队列中的消息按照加权完成传输时间从小到大排列,若加权完成传输时间相等,则按照优先级p_i从高到低排序. 通过这种混杂切换调度模型,能够有效兼顾消息优先级与时间要求,从而满足机器人网络中不同类型数据的多样化传输需求.

4.2　混杂切换模型的切换规则

基于混杂切换调度模型和调度参数,定义子系统切换规则如下.

(1)时间优先子系统切换到优先级优先子系统的条件:当前待发送队列的平均已等待时间$\bar{t}$满足$\bar{t}>t_{\max}$且高优先级数据拥塞时切换;或当前待发送队列的平均已等待时间$\bar{t}$满足$t_{\min}<\bar{t}<t_{\max}$且队列中不存在即将超时数据时切换;或当$\bar{t}<t_{\min}$时切换,此时需进一步判断优先级高于时间敏感消息数据队列中数据的剩余等待时间是否充足,若充足,则将时间敏感类型数据优先级调整为最高,其余按优先级从高到低传输,否则,按原优先级从高到低顺序传输. 该切换策略在确保高优先级数据具有充足传输时间的前提下,通过动态调整时间敏感数据的优先级,实现了对时间敏感类型数据的传输优化.

(2)优先级优先子系统切换到时间优先子系统的条件:当前待发送队列的平均已等待时间$\bar{t}$大于剩余等待时间上界$t_{\max}$且高优先级数据不拥塞时切换;或者当$t_{\min}<\bar{t}<t_{\max}$且队列中存在即将超时数据时切换. 并且当$\bar{t}<t_{\min}$时,需要再次判断优先级高于时间敏感消息数据队列中数据的剩余等待时间是否充足,决定是否调整时间敏感类型数据优先级.

4.2.1　剩余等待时间上界和下界

剩余等待时间上界和下界$t_{\max}$和$t_{\min}$定义如下:

$$t_{\max}=\sum_{i=0}^{N-1}r_{\max}(p_i)\left(\text{transmax}(p_i)-\frac{\text{RTT}}{2}\right)\cdot\frac{n_{p_i}}{\sum_{i=0}^{N-1}n_{p_i}} \tag{12}$$

$$t_{\min}=\sum_{i=0}^{N-1}r_{\min}(p_i)\left(\text{transmax}(p_i)-\frac{\text{RTT}}{2}\right)\cdot\frac{n_{p_i}}{\sum_{i=0}^{N-1}n_{p_i}} \tag{13}$$

其中,n_{p_i}表示队列中优先级为p_i的消息数量;RTT为往返时延;$r_{\max}(p_i)$和$r_{\min}(p_i)$分别表示优先级为p_i的剩余等待时间上界系数和下界系数.

4.2.2　上界系数计算

对于$r_{\max}(p_i)$,当$p_{\text{time}}<p_i\leqslant p_0$时:

$$r_{\max}(p_i)=r_{\max}(p_{\text{time}})+(i+1)\frac{r_{\max}(p_{N-1})-r_{\max}(p_{\text{time}})}{N-1} \tag{14}$$

当$p_{N-1}<p_i<p_{\text{time}}$时:

$$r_{\max}(p_i)=r_{\max}(p_{N-1})-(N-1-i)\frac{r_{\max}(p_{N-1})-r_{\max}(p_{\text{time}})}{N-1} \tag{15}$$

其中,$r_{\max}(p_{\text{time}})$和$r_{\max}(p_{N-1})$分别表示最小上界系数和最大的上界系数,取值范围为$(0.5,1)$,$r_{\max}(p_i)$越小,表示该类数据对时间的敏感度越高. 本研究中,时间敏感类型数据对时间要求较为苛刻,被设定为最小值,这确保了时间敏感数据对总体剩余等待时间上界的降低贡献最大,从而提高系统对时间敏感数据的响应能力,其他数据类型对时间要求的苛刻程度按照优先级从高到低依次下降.

4.2.3　下界系数计算

对于$r_{\min}(p_i)$,当$p_{\text{time}}<p_i\leqslant p_0$时:

$$r_{\min}(p_i)=r_{\min}(p_{\text{time}})+(i+1)\frac{r_{\min}(p_{N-1})-r_{\min}(p_{\text{time}})}{N-1} \tag{16}$$

当$p_{N-1}<p_i<p_{\text{time}}$时:

$$r_{\min}(p_i)=r_{\min}(p_{N-1})-(N-1-i)\frac{r_{\min}(p_{N-1})-r_{\min}(p_{\text{time}})}{N-1} \tag{17}$$

其中,$r_{\min}(p_{\text{time}})$和$r_{\min}(p_{N-1})$分别表示最小下界系数和最大的下界系数,取值范围为$(0,0.5)$,$r_{\min}(p_i)$越小,表示该类数据对时间的敏感度越高,优化系统对时间敏感数据的响应能力.

4.2.4 超时判定

当待发送队列中的消息数据满足以下条件时，判定该消息数据即将超时：

$$t_w > r_e(p_i)\cdot\left(\text{transmax}(p_i)-\frac{\text{RTT}}{2}\right) \tag{18}$$

其中，$r_e(p_i)$为超时系数. 其计算方法如下：

当$p_{\text{time}}<p_i\leqslant p_0$时：

$$r_e(p_i)=r_e(p_{\text{time}})+(i+1)\frac{r_e(p_{N-1})-r_e(p_{\text{time}})}{N-1} \tag{19}$$

当$p_{N-1}<p_i<p_{\text{time}}$时：

$$r_e(p_i)=r_e(p_{N-1})-(N-1-i)\frac{r_e(p_{N-1})-r_e(p_{\text{time}})}{N-1} \tag{20}$$

其中，$r_e(p_{\text{time}})$和$r_e(p_{N-1})$分别表示最小和最大超时系数，取值范围为$(0,1)$.

4.2.5 优先级高于时间敏感数据等待时间判定

对于优先级高于时间敏感消息数据队列，当存在消息数据满足：

$$t_w > r_{\min}(p_i)\cdot\left(\text{transmax}(p_i)-\frac{\text{RTT}}{2}\right)(p_i>p_{\text{time}}) \tag{21}$$

则判定队列中存在高优先级消息数据剩余等待时间不充足.

4.2.6 高优先级数据拥塞判定

若当前待发送队列消息数据满足：

$$n_{p_0}\geqslant r_m\cdot\sum_{i=0}^{N-1}n_{p_i} \tag{22}$$

则可判定当前队列中高优先级数据处于积压状态，需优先调度以缓解高优先级数据传输压力. 其中，r_m为拥塞系数，取值范围为$(0,1)$，表示高优先级数据在待发送队列中的占比. n_{p_i}表示优先级为p_i的消息数据的数量.

4.3 调度算法

在上述混杂切换调度模型的基础上，设计了DRECHS调度算法. DRECHS调度算法根据队列中消息数据平均等待时间$\bar{t}$、单个数据已等待时间t_w、不同优先级的最大允许传输时间$\text{transmax}(p_i)$以及剩余等待时间上下界$t_{\max}$、$t_{\min}$等系统状态量以及切换规则来判断是否切换当前子系统. 如果满足切换条件，进行子系统切换，然后发送子系统待发送队列的队首数据；如果不满足切换条件，直接发送子系统待发送队列的队首数据. DRECHS算法如算法1所示. 其中，依据式(18)判定队列中是否存在超时数据，依据式(21)判定高优先级消息等待时间是否充足，依据式(22)判定高优先级数据是否拥塞.

“优先级优先”侧重于数据本身的业务重要性，而“时间优先”侧重于数据的传输紧迫性. 在网络资源受限时，若严格执行“优先级优先”，可能导致较低优先级且超时的“时间敏感数据”被饿死，从而违背系统实时性需求；若严格执行“时间优先”，则可能导致核心业务的高优先级数据因频繁让路而造成任务失败或推理延迟过高. 冲突的本质是系统服务质量中可靠性与实时性之间的冲突.

本文提出的DRECHS算法为了解决这一冲突，通过混杂切换系统在两个子系统间动态切换来实现权衡. 当系统检测到高优先级数据拥塞且整体等待时间过长时，系统判定此时“可靠性需求”强于“实时性需求”，强制切换回优先级优先子系统，防止关键任务积压. 当系统检测队列中出现超时数据且整体等待时间在容忍范围内时，系统判定此时“实时性需求”强于“可靠性需求”，切换回时间优先子系统，优先处理紧迫数据.

对算法1的时间和空间复杂度进行如下分析. 对于待发送消息数据量为n的情况下，DRECHS算法的时间复杂度为$O(n^2)$，空间复杂度为$O(n)$.

证明 时间复杂度分析.

算法1 DRECHS调度算法

```
输入: 待发送消息数据队列
输出: 待发送队列中的队首消息数据
WHILE TRUE DO
  IF 当前处于时间优先子系统 THEN
    IF t̄ > t_max and 高优先级数据拥塞 THEN
      切换至优先级优先子系统
    ELSIF t_min < t̄ < t_max and 不存在超时数据
THEN
      切换至优先级优先子系统
    ELSIF t̄ < t_min THEN
      IF 高优先级数据等待时间充足 THEN
        提升时间敏感型数据优先级
      END IF
      切换至优先级优先子系统
    END IF
ELSIF 当前处于优先级优先子系统 THEN
  IF t̄ > t_max and 高优先级数据不拥塞THEN
    切换至时间优先子系统
  ELSIF t̄ < t_min THEN
    IF 高优先级数据等待时间充足 THEN
      提升时间敏感型数据优先级
    END IF
  ELSIF t_min < t̄ < t_max and 存在超时数据
THEN
      切换至时间优先子系统
    END IF
  END IF
  获取队首数据
END WHILE
```

依据优先级优先队列构建时间优先队列的过程采用插入排序算法. 时间队列的长度在插入过程中逐步增加,对于第i个元素的插入.

(1)当$i=1$时,不需要比较,时间复杂度为$O(1)$;

(2)当$i>1$时,需要比较$i-1$次,时间复杂度为$O(i-1)$.

因此,对于n个元素,总的插入时间复杂度为$O(1+1+2+3+\cdots+n-1)=O\left(\frac{n(n-1)}{2}\right)=O(n^2)$.

算法中的其他操作包括:

(1)计算t_{max}和t_{min}:遍历优先级优先队列,复杂度$O(n)$;

(2)计算平均等待时间和检查超时消息:遍历时间优先队列,复杂度$O(n)$;

(3)检查高优先级数据等待时间:遍历高优先级队列(消息个数$\leqslant n$),复杂度$O(n)$.

综上,算法的主导复杂度来自时间优先队列的构建过程,总体时间复杂度为$O(n^2)$.

空间复杂度分析如下:

(1)优先级优先队列:存储n个消息,空间复杂度$O(n)$;

(2)时间优先队列:存储n个消息,空间复杂度$O(n)$.

各队列的空间复杂度线性叠加,因此算法的总体空间复杂度为$O(n)$.

证毕.

5　实验分析

为了全面评估DRECHS调度算法的性能,本研究将其集成实现到ROS2的流控制器(FlowController)模块中,开展了仿真实验和实物测试. 两类实验均采用Neurosurgeon的动态分区策略:通过测量端边设备之间的带宽,计算每层作为分割点时的总耗时,选择总耗时最小的层作为分割点.

在仿真实验中,通过本地Linux系统模拟端边协同场景,对比DRECHS算法与ROS2中DDS的流控制器当中内置调度算法的性能. 由于Neurosurgeon在不同带宽条件下对应不同的分割点和中间传输数据量,因此实验系统地分析了各带宽条件下不同优先级数据的传输延迟和丢包率.

在实物测试中,将集成DRECHS策略的ROS2部署在Nvidia Xavier NX(移动端)和PC(边缘服务器)上,构建了基于Neurosurgeon的端边协同测试环境. 考虑到带宽测试数据通过DDS传输可能导致测量精度较低,本研究采用节点解耦的分层通信部署方案:将分区算法独立部署为专门节点,使用socket负责进行带宽测试和分割点计算;端边协同节点仅负责接收分割点信息并执行DNN推理任务.

5.1　仿真实验设计与分析

5.1.1　仿真实验设计

仿真实验场景假设为移动机器人与边缘服务器进行端边协同推理,将DRECHS算法集成到ROS2的流控制器模块当中,推理采用AlexNet神经网络,模拟混合数据流的通信场景.

仿真实验只考虑理论分析中提到的3种类型的数据,发布者负责生成并发布3种不同优先级的数据,发布总数固定为5 000组,订阅者负责接收数据并计算出每种优先级数据的传输延迟和丢包率. 实验中,在不同的带宽条件和推理任务数据的不同发布周期下,将本算法与流控制器当中内嵌的3种调度策略FIFO、ROUNDROBIN和HIGHPRIORITY进行对比,此外,文献[25]将GEDF(Global Earliest Deadline First)算法核心思想实现在ROS2执行器当中[25],受此工作启发,本研究将截止时间优先典型算法GEDF核心思想集成在ROS2-DDS中进行对比,以及与面向数据延迟和数据大小的KFF调度算法[27]的核心调度思想进行对比,评估不同优先级数据的传输延迟和丢包率.

其中,FIFO基于数据的到达顺序进行调度;ROUNDROBIN算法追求公平性,依次调度不同的数据流;HIGHPRIORITY是基于优先级的调度算法,优先处理优先级高的数据,对高优先级数据的优化效果显著;GEDF是基于时间属性的调度算法,优先处理接近截止时间的任务;KFF调度算法依据数据流的服务质量需求和流量大小分配数据优先级,优先处理优先级高的数据,并依据已发送字节数和降级阈值动态更新数据流优先级. 推理任务中间数据量的确定采用Neurosurgeon神经网络分区算法的思想,因此不同带宽条件对应不同的最优分割点和数据量. 每种优先级消息数据的具体配置情况详见表2和表3.

表2　消息数据配置

数据类型	优先级	数据量	发布周期/s
任务关键型	0(高)	动态变化(见表3)	0.01/0.05/0.09
时间敏感型	1	2 000	0.03
普通关键型	6(低)	1×10^6	0.07

表3　AlexNet网络不同分割点对应数据量

带宽/mbps	分割点/层	传输向量格式	数据量/Byte
100	0	(1,3,227,227)	618 348
80	2	(1,96,27,27)	279 936
60	4	(1,256,13,13)	173 056
40	4	(1,256,13,13)	173 056
20	12	(1,2)	8

5.1.2 仿真实验结果与分析

本研究针对DRECHS、FIFO、ROUNDROBIN、HIGHPRIORITY、GEDF和KFF这6种调度算法，在不同带宽条件和推理任务数据发布周期下进行了系统性对比实验. 为确保实验结果的可靠性与代表性，每组实验均进行多次重复测试并取平均值. 实验重点分析了各算法对不同优先级数据的传输性能表现.

对于最高优先级的DNN推理任务中间数据，传输可靠性是核心性能指标. 如图5~图7所示，分别展示了在不同推理任务发布周期和网络带宽的条件下，不同算法的丢包率对比，从图中可以看出，DRECHS算法在任务关键型数据的丢包率表现优于FIFO、ROUNDROBIN、GEDF和KFF算法，与专门针对高优先级数据优化的HIGHPRIORITY算法相近. 这充分验证了DRECHS算法在保障任务关键型数据传输可靠性方面的有效性. 图8展示了不同推理任务发布周期和网络带宽的条件下，不同算法任务关键型数据的传输延迟表现，DRECHS算法对任务关键型数据的传输延迟处理略逊于HIGHPRIORITY算法，与优化延迟敏感流传输延迟性能较为优秀的KFF算法相近，同时明显优于其他3种对比算法，满足应用需求.

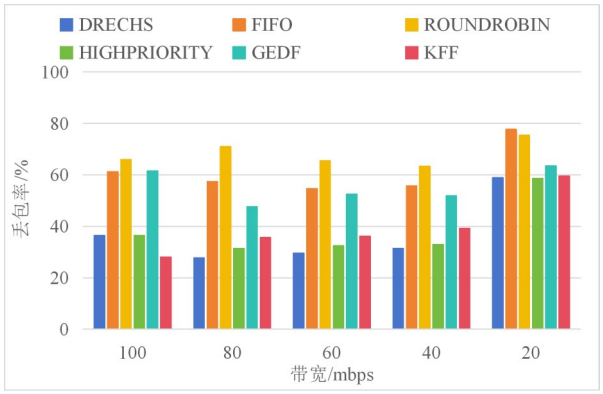


图5 推理任务数据发布周期0.01 s下任务关键型消息数据丢包率

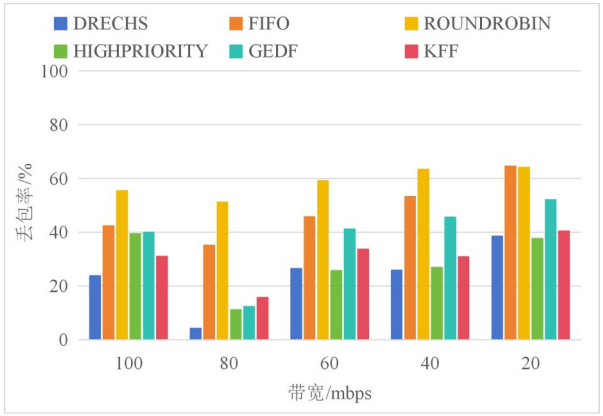


图6 推理任务数据发布周期0.05 s下任务关键型消息数据丢包率

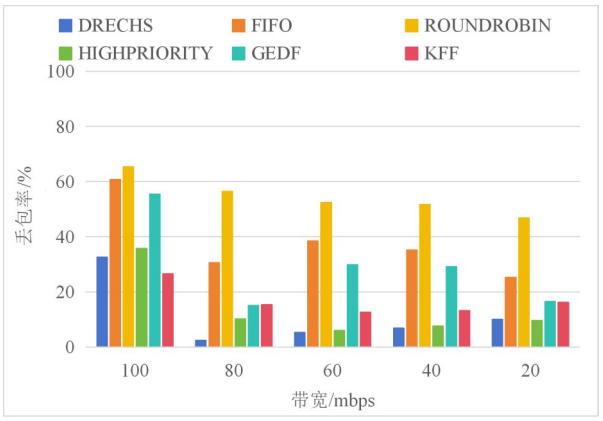


图7 推理任务数据发布周期0.09 s下任务关键型消息数据丢包率

从100 mbps到20 mbps的实验结果显示，高优先级数据丢包率呈现先下降后上升的“U型”趋势. 在100 mbps高带宽条件下，对应的高优先级数据量较大，尽管带宽充足，但巨大的数据量导致网络传输压力显著增大，进而引起较高的丢包率；在40~80 mbps范围内，分割点移至网络中间层，传输数据量显著减少，使丢包率有所降低；而在20 mbps低带宽条件下，尽管选择了网络末端层作为分割点，传输数据量仅为8字节，但带宽资源严重受限成为决定性因素，导致传输拥塞和丢包率再次明显上升.

时间敏感消息数据作为次高优先级数据，主要包含机器人位置信息等关键状态数据，对传输可靠性和传输延迟均有较高要求. 实验结果表明，在不同推理任务数据发布周期和带宽条件下，DRECHS算法对时间敏感消息数据的传输性能具有显著改善.

对于次高优先级的时间敏感消息，实验重点考察其可靠性和传输延迟指标. 如图9~图11所示，分别展示了在不同推理任务发布周期和网络带宽的条件下，不同算法的丢包率对比，DRECHS算法在各种工作条件下均改善了时间敏感消息的传输性能. 具体而言，其丢包率普遍优于FIFO、ROUNDROBIN、HIGHPRIORITY和GEDF算法，与对延迟敏感流优化性能较好的KFF算法基本相近. 如图12所示，展示了不同推理任务发布周期和网络带宽的条件下，不同算法时间敏感数据的传输延迟表现，从图中可以看出，DRECHS算法的传输延迟与HIGHPRIORITY和KFF算法基本相近，同时优于其他3种调度算法.

对于低优先级的普通关键消息，此类消息普遍对时延要求比较宽松. 如图13所示，在3种推理任务数据发布周期下，计算了每种算法所有带宽情况的平均传输延迟，DRECHS算法对低优先级数据的平均传输时延普遍高于其他对比算法. 这是由于算法通过适度调节低优先级且时间不敏感数据的传输效率，以确保高优先级和时间敏感数据的优质传输. 图14~图16分别

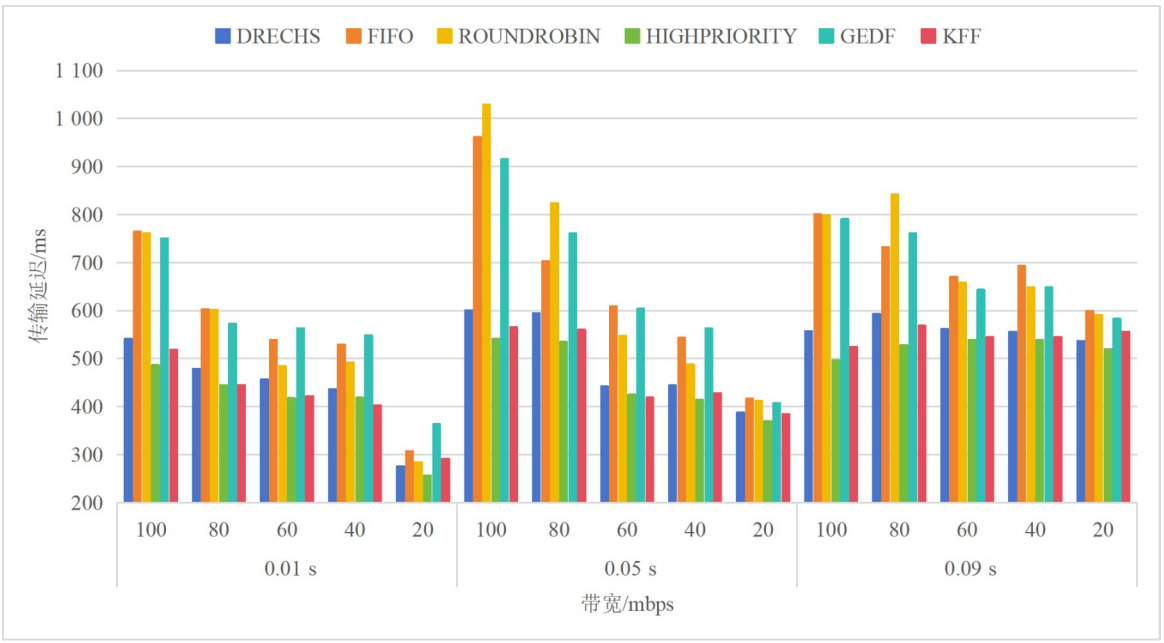


图8　任务关键型消息数据传输延迟

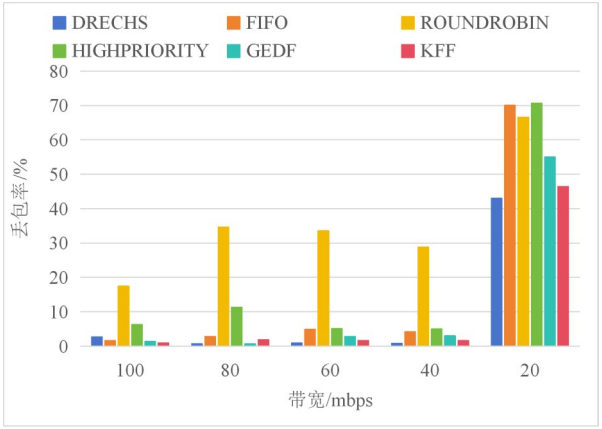


图9　推理任务数据发布周期0.01 s下时间敏感消息数据丢包率

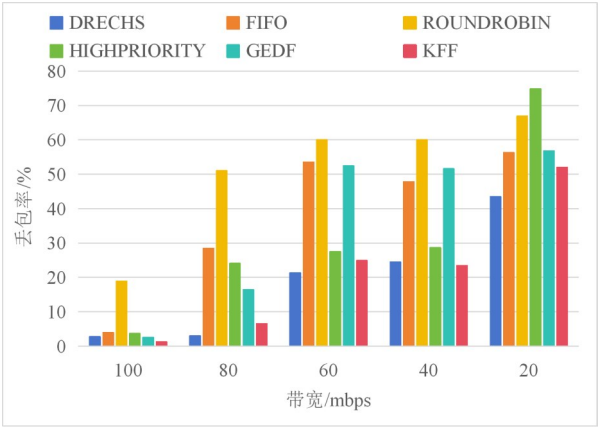


图11　推理任务数据发布周期0.09 s下时间敏感消息数据丢包率

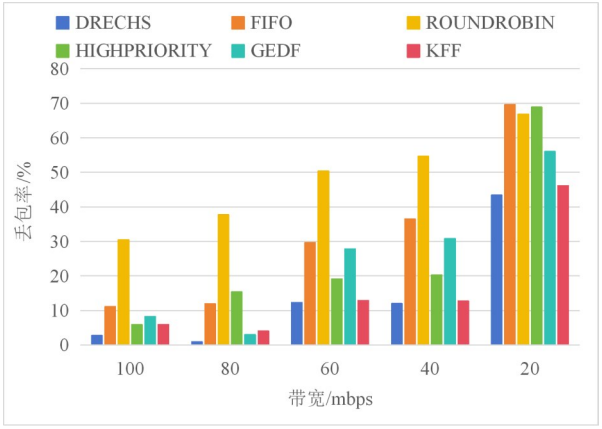


图10　推理任务数据发布周期0.05 s下时间敏感消息数据丢包率

展示了在不同推理任务发布周期和网络带宽的条件下不同算法的普通关键型消息的丢包率对比,虽然FIFO和GEDF算法在低优先级消息的丢包率表现最优,但DRECHS算法仍优于HIGHPRIORITY算法,表明其在低优先级消息传输可靠性方面具有一定优化效果.

深入分析图14~图16中带宽对算法性能的影响发现,在20~80 mbps带宽范围内,DRECHS算法的丢包率低于ROUNDROBIN算法. 然而,当带宽为100 mbps时,DRECHS算法和HIGHPRIORITY算法的丢包率出现显著上升,这是由于高带宽条件下系统中高优先级数据流量增大,两种算法为保证高优先级数据的传输性能,不可避免地影响了低优先级数据的传输可靠性. 然而,在这种情况下,DRECHS算法对低优先级数据的处理性能仍然优于专门针对高优先级数据优化的HIGHPRIORITY算法. 这一现象反映了DRECHS算法在保障高优先级数据传输质量的同时,通过混杂切换模型的动态调度机制,缓解了对低优先级数据的“饥

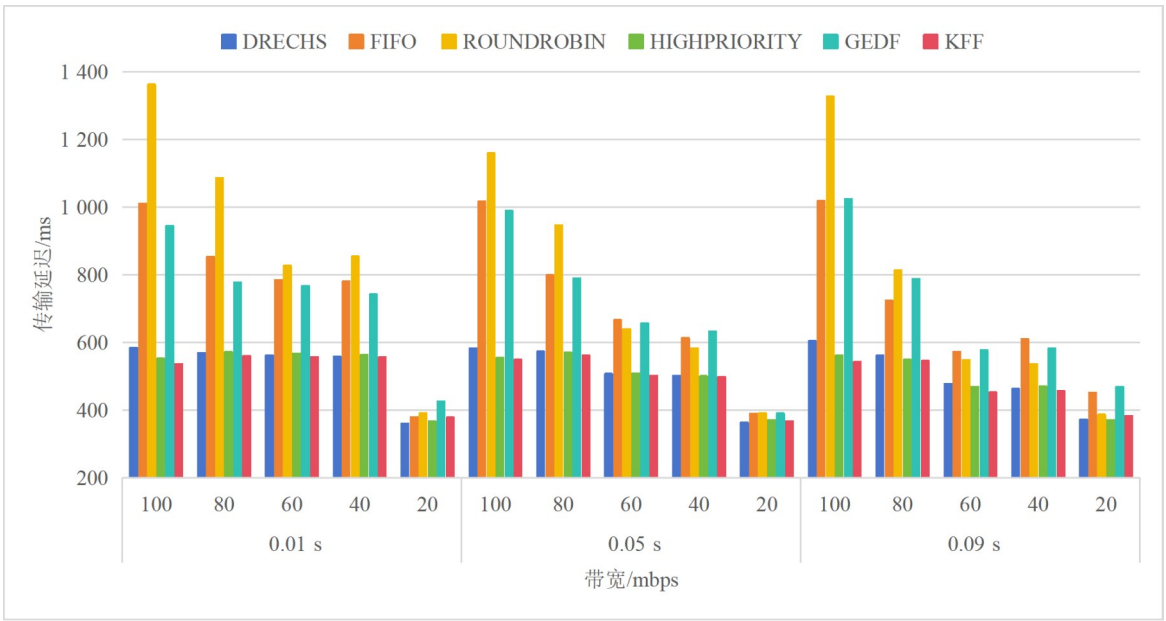


图12 时间敏感消息数据延迟

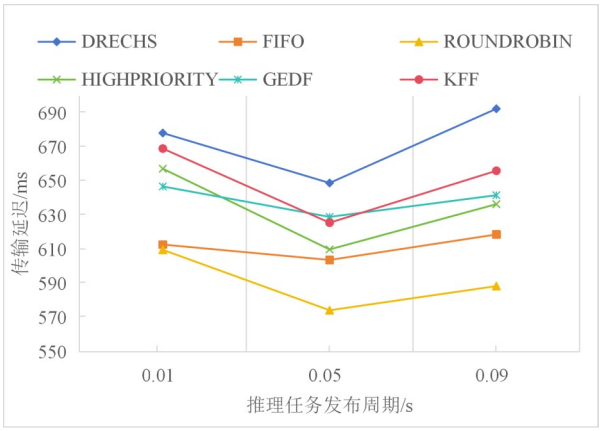


图13 普通关键消息数据延迟

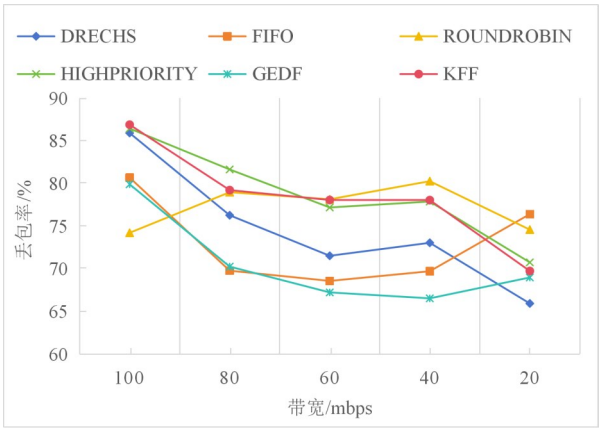


图15 推理任务数据发布周期0.05 s下普通关键消息丢包率

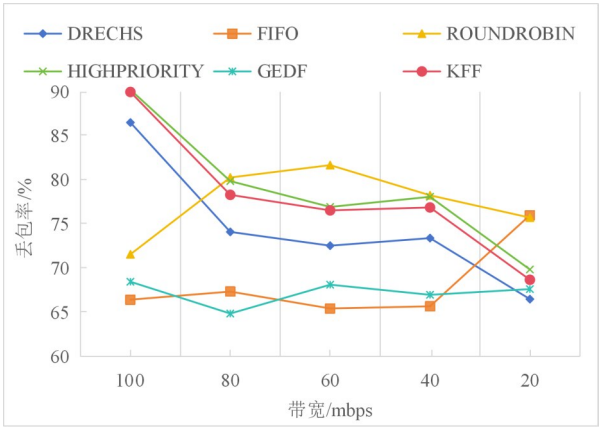


图14 推理任务数据发布周期0.01 s下普通关键消息丢包率

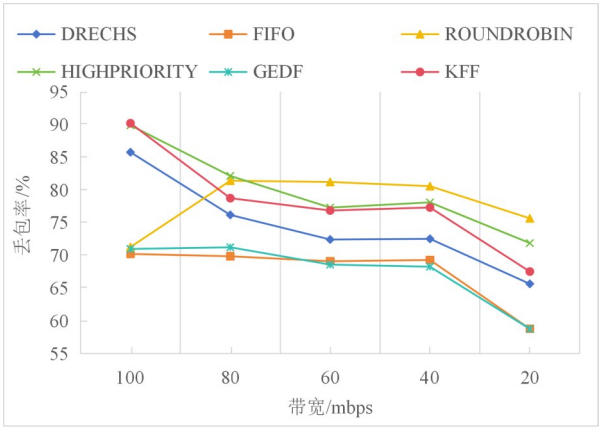


图16 推理任务数据发布周期0.09 s下普通关键消息丢包率

饿”现象.

如图17所示,在3种推理任务数据发布周期下,计算了每种算法所有带宽情况的平均传丢包率. 结果显示,DRECHS算法的总体丢包率最低,表明DRECHS通过动态调度策略实现了对各优先级数据流的差异化服务质量优化,在保证任务关键型数据可靠传输的同时,

显著改善了时间敏感数据的传输性能,并保持了普通关键数据的传输可靠性,最终实现了系统整体传输性能的提升.

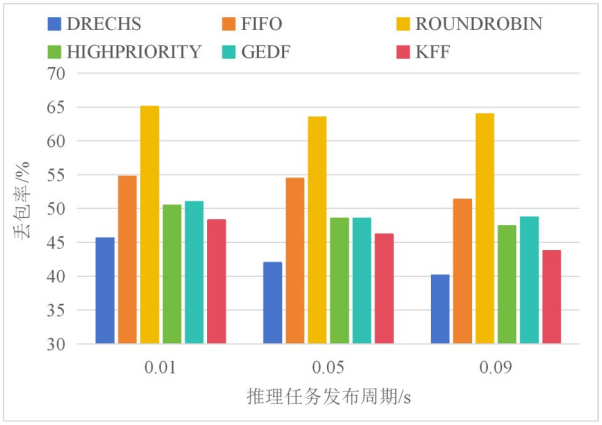


图 17　不同推理任务数据发布周期总丢包率

为验证DRECHS算法在网络带宽突变条件下的有效性,本研究设计了带宽突变环境下的对比实验. 实验过程中,网络带宽从初始的 100 mbps 降至 60 mbps,随后进一步降至 20 mbps,模拟了实际网络环境中可能出现的带宽突变场景. 在此环境下,本研究对比分析了DRECHS算法与现有调度算法在处理不同优先级消息时的丢包率表现.

图 18~图 21 分别展示了在不同推理任务发布周期和突变带宽条件下,推理任务、时间敏感、普通关键消息以及全部消息数据不同算法的丢包率情况. 实验结果显示,对于推理任务消息数据,DRECHS 的丢包率与 HIGHPRIORITY 算法相近,但明显低于其他对比算法;在处理普通关键消息时,DRECHS 的性能优于 HIGHPRIORITY 算法;而对于时间敏感消息数据以及全部消息的总体丢包率指标,DRECHS 算法均实现了相对最低的丢包率.

综合以上实验数据分析,可以得出结论:DRECHS 调度算法在网络带宽突变这一关键场景下,相较于对

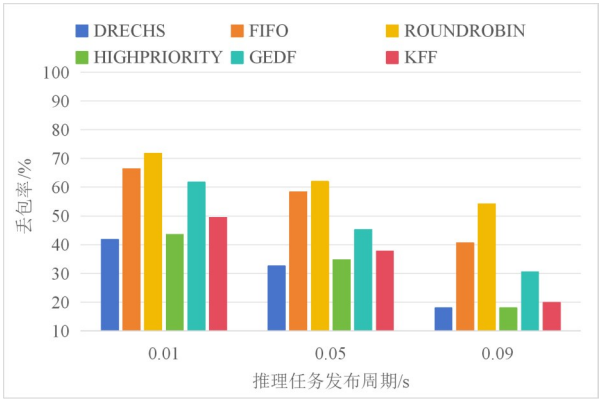


图 18　推理任务消息数据丢包率

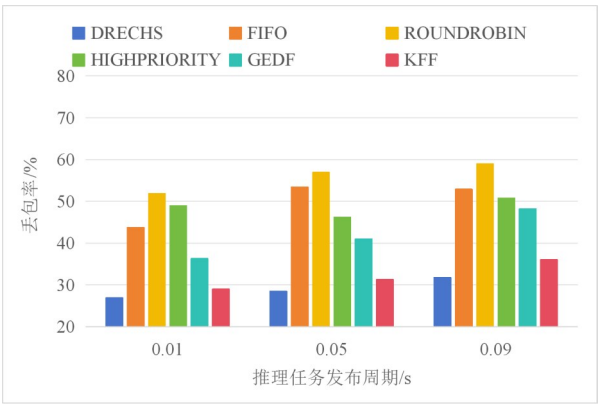


图 19　时间敏感消息数据丢包率

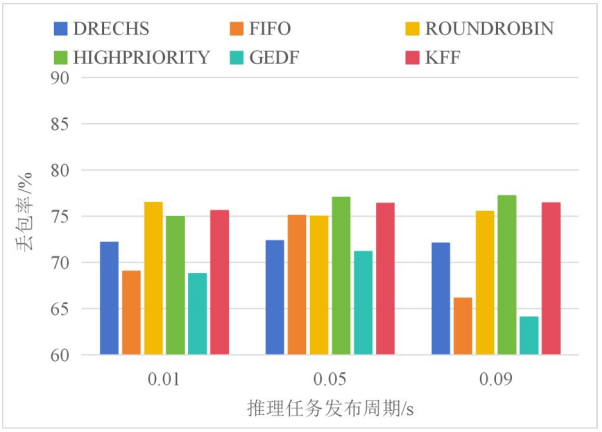


图 20　普通关键消息数据丢包率

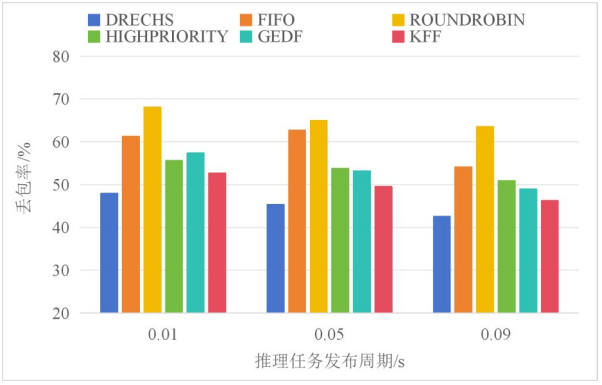


图 21　3 种优先级消息数据总丢包率

比算法具有更为稳健的消息传输可靠性和更优的差异化服务质量保障能力. 这一结果验证了该算法对网络资源动态变化的良好适应性.

为验证优先级优先子系统和时间优先子系统的必要性,本文设计了对比实验. 由于DRECHS的优先级子系统与HIGHPRIORITY算法思想相同,且前文已证明DRECHS相较于HIGHPRIORITY具有一定的改进,因此本部分重点验证优先级优先子系统的必要性. 当高优先级数据量较大时,队列中会出现高优先级数据拥塞现象. 根据式(22),系统会切换至优先级优先子系

统,以保障高优先级数据的传输质量,因此,实验环境设置为高优先级数据量较大的条件下,对比DRECHS与仅采用时间优先子系统的性能表现.如图22所示,DRECHS降低了高优先级数据的传输延迟.图23展示了两种方法的丢包率对比,结果表明两种方法在高优先级数据丢包率上基本持平.综上所述,DRECHS能够在保持高优先级数据丢包率的前提下,降低传输延迟,从而验证优先级优先子系统的必要性.

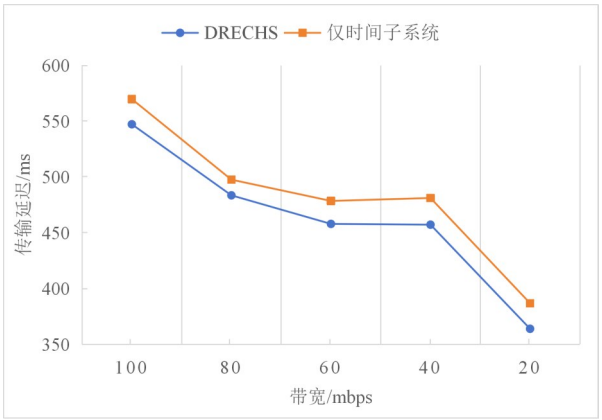


图22 高优先级数据传输延迟

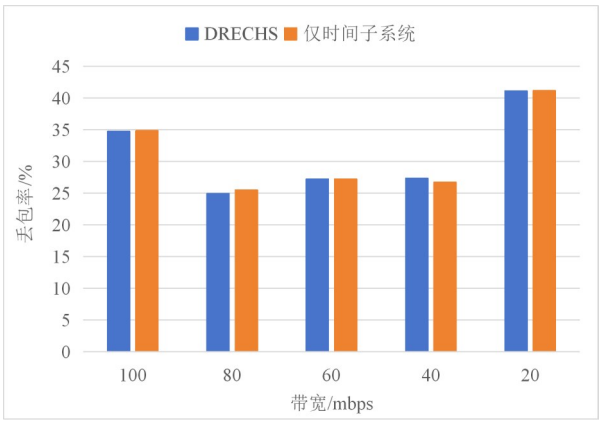


图23 高优先级数据丢包率

综上所述,通过系统的对比实验验证,DRECHS算法成功实现了面向不同优先级数据流的服务质量差异化优化.该算法不仅在保障任务关键型数据传输可靠性方面表现突出,而且实现了与时间敏感数据传输需求的良好平衡,同时在保持普通关键数据传输可靠性方面也取得了一定的优化效果,最终实现了全局数据传输性能的整体提升.

5.2 实物应用与测试

为了验证DRECHS调度算法在实际环境中对3种类型混合数据流的调度性能,本研究将集成了DRECHS调度策略的ROS2分别部署在机器人搭载的Nvidia Xavier NX(移动端)和PC(边缘服务器)上.移动端和边缘设备主要规格对比如表4所示.同时,基于Neurosurgeon的算法思想构建了基于AlexNet的端边协同图像分类系统.在移动端部署的ROS2发布者节点负责接收分割点信息并执行神经网络前段推理;边缘服务器部署的ROS2订阅者节点负责接收混合数据流并完成后段推理.

表4 移动端和边缘设备主要规格

规格 类别	Nvidia Xavier NX	PC
CPU	6核ARM(6 MB L2 + 4 MB L3)	10核Intel i5-13400F(20 MB L3)
GPU	NVIDIA Volta(384核)	Nvidia GeForce RTX 4060 Ti
内存	8/16 GB LPDDR4x	32 GB(16 GB*2)DDR5
存储	16 GB eMMC 5.1	1 TB

基于Neurosurgeon的动态分区方法在DNN端边协同推理中要求移动端执行分割点计算和推理任务,其中,分割点的计算依赖带宽测试结果.在网络通信架构中,DDS作为位于应用层和传输层之间的中间件,提供了丰富的数据分发服务,适用于需要差异化服务质量的混合数据流的传输.然而,对于带宽测试这类需要精确测量的维护数据,DDS的复杂抽象机制反而会影响其准确性.在实际机器人系统部署中,通常会根据不同数据的传输需求选择合适的通信方式.因此,在本研究中,将Neurosurgeon分区算法独立部署为专门的ROS2节点,通过socket在传输层直接完成带宽测试;而将需要区分服务质量的数据流通过DDS传输,由DRECHS算法进行调度优化.这种通信方式选择使带宽测试能够避免中间件开销获得较高的测量精度,同时确保了对关键业务数据流的调度优化.

在此部署方案下,移动端的端边协同节点通过DDS中间件层专注于执行DNN前段推理和混合数据流的发布任务.运行结果表明,这种部署架构的设计在保证了带宽测量精度的同时,还能高效获取分割点信息,显著提升了系统运行效率.

系统部署架构如图24所示,包含4个核心节点,节点1和节点3部署在移动端,节点2和节点4部署在边缘服务器.其中,节点1和节点2负责分割点的计算和更新,节点3和节点4执行端边协同推理任务,处理多种类型的混合数据流.具体运行流程如下.

(1)节点1通过socket周期性(数秒间隔)发送带宽测试信息至节点2.

(2)节点2基于带宽数据计算最优分割点,并通过ROS2 publisher发布至节点3.

(3)节点3执行DNN前段推理并通过DDS发布中间结果及其他类型数据.当接收到新的分割点信息时,先更新分割点再执行推理任务.

(4)节点4的多个订阅者接收不同类型的数据流,

完成DNN后段推理,输出分类结果.

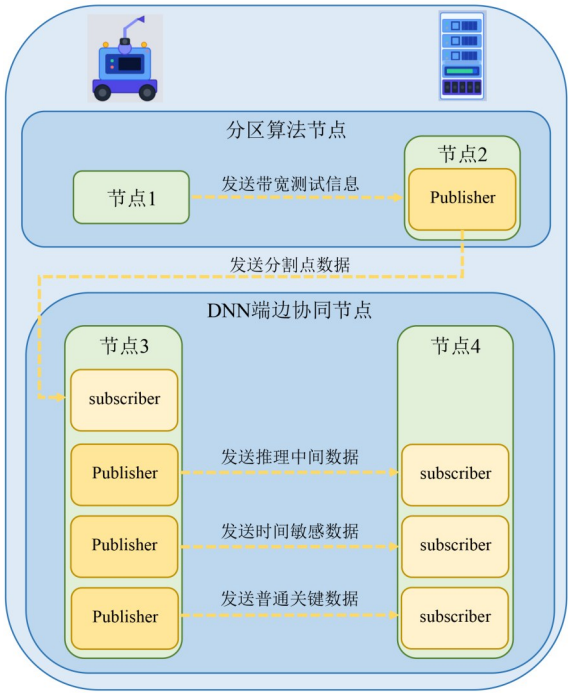


图24　实物部署框架

部署实验验证表明,集成DRECHS调度策略的端边协同系统展现出良好的稳定性和可靠性:移动端能够准确接收分割点信息,生成并发布包含推理中间结果的混合数据流;边缘服务器能够可靠接收并处理多种类型数据流,准确完成推理任务并输出分类结果.

6　结束语

随着5G和人工智能技术的飞速发展,端边云协同框架已成为机器人应用的重要趋势.本文聚焦于基于ROS2通信的机器人DNN端边协同推理的混合数据流调度问题,提出了DRECHS调度算法,以优化多数据类型的传输性能并满足端边协同推理对实时性和可靠性的需求.

首先,基于DNN端边协同推理框架的通用基本耗时构成,定义了端边协同任务数据传输的最大允许传输时间边界条件,为混合数据流传输优化提供了理论基础.将上层应用的需求映射到底层通信参数,形成了一个应用需求与通信优化的结合设计.

其次,针对不同类型数据传输需求的差异性,基于混杂切换系统理论构建了包含优先级子系统和时间子系统的调度模型.在此模型基础上,设计并实现了DRECHS混合数据流动态调度算法,该算法通过综合考虑数据优先级和传输紧迫程度,实现了对不同类型数据的高效调度.

最后,实验验证方面,本研究将DRECHS算法集成到ROS2的DDS流控制器中.基于实验选用的动态分区方法,在不同带宽条件下通过仿真实验系统地评估了算法性能.实验结果表明,DRECHS算法通过混杂切换系统模型和动态调度策略,实现了对不同优先级数据流的差异化服务质量优化,在满足高优先级数据传输需求的同时,显著提升了系统整体传输性能,在传输延迟和丢包率等关键指标上具有一定优势.并在真实设备上部署了DRECHS调度算法以及DNN端边协同推理框架,验证了该方案在实际应用环境中的可行性和有效性.

未来工作中,可针对多机器人系统中的混合数据流调度问题,进一步探索构建带约束的数学优化模型,利用混合整数非线性规划(Mixed-Integer NonLinear Programming,MINLP)进行建模并求解调度,同时可进一步探索在更复杂的应用场景下优化混合数据流传输性能,并研究如何将本文提出的方法扩展到更广泛的端边云协同应用中.

参考文献

[1] 李智勇, 王琦, 陈一凡, 等. 车辆边缘计算环境下任务卸载研究综述[J]. 计算机学报, 2021, 44(5): 963-982.
LI Z Y, WANG Q, CHEN Y F, et al. A survey on task offloading research in vehicular edge computing[J]. Chinese Journal of Computers, 2021, 44(5): 963-982. (in Chinese)

[2] 周悦芝, 张迪. 近端云计算: 后云计算时代的机遇与挑战[J]. 计算机学报, 2019, 42(4): 677-700.
ZHOU Y Z, ZHANG D. Near-end cloud computing: Opportunities and challenges in the post-cloud computing era[J]. Chinese Journal of Computers, 2019, 42(4): 677-700. (in Chinese)

[3] GROSHEV M, BALDONI G, COMINARDI L, et al. Edge robotics: Are we ready? An experimental evaluation of current vision and future directions[J]. Digital Communications and Networks, 2023, 9(1): 166-174.

[4] CHOI H S, ENRIGHT D, SOBHANI H, et al. Priority-driven real-time scheduling in ROS 2: Potential and challenges[C]//1st International Workshop on Real-time And intelliGent Edge computing (RAGE). Piscataway: IEEE, 2022.

[5] 芦倩, 李晓娟, 关永, 等. 面向数据流的ROS2数据分发服务形式建模与分析[J]. 软件学报, 2021, 32(6): 1818-1829.
LU Q, LI X J, GUAN Y, et al. Modeling and analysis of ROS2 data distribution service for data flow[J]. Journal of Software, 2021, 32(6): 1818-1829. (in Chinese)

[6] 鲁敬敬, 秦云川, 刘志中, 等. 机器人操作系统ROS安全性研究综述[J]. 软件学报, 2024, 35(2): 1010-1027.
LU J J, QIN Y C, LIU Z Z, et al. Survey on security of robot operating system ROS[J]. Journal of Software, 2024, 35(2): 1010-1027. (in Chinese)

[7] SABOIA M, CLARK L, THANGAVELU V, et al. ACHORD: Communication-aware multi-robot coordination with intermittent connectivity[J]. IEEE Robotics and Automation Letters, 2022, 7(4): 10184-10191.

[8] MAHEMUTI P, YANG L, LUO L L. Modeling, analysis and synthesis of hybrid system: A review[J]. Applied Mechanics and Materials, 2014, 615: 36-43.

[9] 陈阳, 皮德常, 代成龙, 等. 多无人机协同陆地设施辅助移动边缘计算的系统能耗最小化方法[J]. 电子学报, 2023, 51(4): 984-992.
CHEN Y, PI D C, DAI C L, et al. Energy minimization for multi-UAVs cooperative ground access points assisted mobile edge computing[J]. Acta Electronica Sinica, 2023, 51(4): 984-992. (in Chinese)

[10] 许小龙, 杨威, 杨辰翊, 等. 车联网边缘计算环境下基于流量预测的高效任务卸载策略研究[J]. 电子学报, 2025, 53(2): 329-343.
XU X L, YANG W, YANG C Y, et al. Efficient task offloading based on traffic prediction in IoV-enabled edge computing[J]. Acta Electronica Sinica, 2025, 53(2): 329-343. (in Chinese)

[11] QUERALTA J P, LI Q Q, ZHUO Z, et al. Enhancing autonomy with blockchain and multi-access edge computing in distributed robotic systems[C]//2020 Fifth International Conference on Fog and Mobile Edge Computing. Piscataway: IEEE, 2020: 180-187.

[12] WAN S H, GU Z H, NI Q. Cognitive computing and wireless communications on the edge for healthcare service robots[J]. Computer Communications, 2020, 149: 99-106.

[13] MATSUBARA Y, LEVORATO M, RESTUCCIA F. Split computing and early exiting for deep learning applications: Survey and research challenges[J]. ACM Computing Surveys, 2023, 55(5): 1-30.

[14] KANG Y P, HAUSWALD J, GAO C, et al. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge[J]. ACM SIGARCH Computer Architecture News, 2017, 45(1): 615-629.

[15] HU C, BAO W, WANG D, et al. Dynamic adaptive DNN surgery for inference acceleration on the edge[C]//IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. Piscataway: IEEE, 2019: 1423-1431.

[16] JAHIER PAGLIARI D, CHIARO R, MACII E, et al. CRIME: Input-dependent collaborative inference for recurrent neural networks[J]. IEEE Transactions on Computers, 2021, 70(10): 1626-1639.

[17] JALIL A, KOBAYASHI J, SAITOH T. Performance improvement of multi-robot data transmission in aggregated robot processing architecture with caches and QoS balancing optimization[J]. Robotics, 2023, 12(3): 87.

[18] MARUYAMA Y, KATO S, AZUMI T. Exploring the performance of ROS2[C]//2016 International Conference on Embedded Software. Piscataway: IEEE, 2016: 1-10.

[19] KRONAUER T, POHLMANN J, MATTHÉ M, et al. Latency analysis of ROS2 multi-node systems[C]//2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems. Piscataway: IEEE, 2021: 1-7.

[20] JIANG Z Y, GONG Y F, ZHAI J D, et al. Message passing optimization in robot operating system[J]. International Journal of Parallel Programming, 2020, 48(1): 119-136.

[21] WANG Y P, TAN W D, HU X Q, et al. TZC: Efficient inter-process communication for robotics middleware with partial serialization[C]//2019 IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway: IEEE, 2020: 7805-7812.

[22] LUO X T, JIANG X, TANG Y, et al. Analysis and optimization of communication delay in multi-subscriber environments of ROS 2[J]. Journal of Systems Architecture, 2025, 164: 103428.

[23] TEPER H, BETZ T, GÜNZEL M, et al. End-to-end timing analysis and optimization of multi-executor ROS 2 systems[C]//2024 IEEE 30th Real-Time and Embedded Technology and Applications Symposium. Piscataway: IEEE, 2024: 212-224.

[24] CASTILLO-SÁNCHEZ J B, GONZÁLEZ-PARADA E, CANO-GARCÍA J M. Swarm robot communications in ROS 2: An experimental study[J]. IEEE Access, 2024, 12: 142930-142943.

[25] AL ARAFAT A, WILSON K, YANG K C, et al. Dynamic priority scheduling of multithreaded ROS 2 executor with shared resources[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2024, 43(11): 3732-3743.

[26] TEPER H, BELL O, GÜNZEL M, et al. Reconciling ROS 2 with classical real-time scheduling of periodic tasks[C]//2025 IEEE 31st Real-Time and Embedded Technology and Applications Symposium. Piscataway:

IEEE, 2025: 177-189.

[27] TAO X D, QIAN X Y, HAN L, et al. Key flow first prioritized flow scheduling strategy in multi-tenant data centers[J]. IEEE Transactions on Network and Service Management, 2024, 21(3): 3264-3277.

[28] DING N, WANG Y H, ZHAO Y W, et al. Packet scheduling algorithm based on hybrid system theory in the industrial Internet of Things[C]//2021 IEEE International Conference on Smart Internet of Things. Piscataway: IEEE, 2021: 33-39.

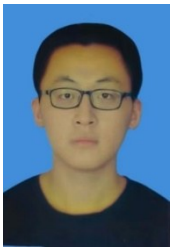
作者简介

丁　男　男,1978年出生于辽宁省沈阳市.现为大连理工大学计算机科学与技术学院教授、博士生导师.主要研究方向为物联网与工业互联网技术、网络系统安全与数据分析、多智能体协同、嵌入式人工智能等.

E-mail: dingnan@dlut.edu.cn

方玺淇　男,2000年出生于黑龙江省齐齐哈尔市.现为大连理工大学计算机科学与技术学院硕士研究生.主要研究方向为云-边-端机器人系统的通信优化、边缘计算.

E-mail: fxq@mail.dlut.edu.cn



郝云涛　男,2002年出生于河北省邢台市.现为大连理工大学计算机科学与技术学院博士研究生.主要研究方向为边缘计算和边缘智能.

E-mail: haoyuntao@mail.dlut.edu.cn

胡创业　男,1994年出生于河南省周口市.现为大连理工大学计算机科学与技术学院博士研究生.主要研究方向为机器人视觉语义导航等.

E-mail: DHCY@mail.dlut.edu.cn

许　力　男,1982年出生于辽宁省大连市.现为字节跳动高级工程师.主要研究方向为网络虚拟化、云计算、概率图模型等.

E-mail: xuli.z@bytedance.com