doi:10.3772/j.issn.1002-0470.2024.09.001

OODAFlow:面向智能无人系统的流式数据处理框架①

全振宇②*** 尹龙祥* 陈晓明* 韩银和*

(*中国科学院计算技术研究所智能计算机研究中心 北京 100190)

(**中国科学院大学 北京 100190)

摘 要 智能无人系统是一种能够在复杂环境中自主进行实时推理、决策和制定行动方案的计算系统。智能无人系统实现实时决策的关键在于对流式数据的实时处理,然而随着人工智能技术和传感器技术的快速发展,智能无人系统需要处理的数据规模不断增长,数据类型变得更加复杂。面对不断增长的数据处理性能需求,智能无人系统需要一个充分优化的专用流式数据处理框架来提升其数据处理性能。针对该问题,本文提出了一种面向智能无人系统的流式数据处理框架 OODAFlow,该框架将智能无人系统的硬件特征和智能计算任务的数据特征与观察-判断-决策-行动(OODA)模型思想相融合,实现了 OODA 任务创建、任务调度、资源调度等功能,能够实现对智能无人系统异构资源的调度和智能计算任务的处理。本文在智能无人系统上搭建了一套 OODA 任务处理系统,验证了所提 OODAFlow 框架的可行性。通过提出的图像预处理过程优化、流水线优化以及判断节点并行加速优化等方法,提高了系统的数据吞吐性能和资源利用率。无人机智能控制任务的实验表明,采用本文提出的 OODAFlow 框架后,智能无人系统的数据处理性能提升了 73 倍。

关键词 智能无人系统;深度学习加速卡;观察-判断-决策-行动(OODA);流式数据处理框架;异构计算资源

无人系统是一种利用硬件计算资源和软件算法实现与周围环境交互,且能够在无人干预的情况下独立执行一项或多项计算任务的计算系统。以深度学习为代表的人工智能技术的快速发展,为无人系统的发展提供了新的机遇。人工智能技术能够为无人系统提供复杂场景下的自主作业能力,有效提高无人系统在数据处理方面的能力。随着人工智能技术和无人系统的深入融合,智能无人系统应运而生。智能无人系统是一种能够自主进行推理、决策,最终形成行动方案的计算系统[1],具有高智能性和自主性,能够适应多变的环境和需求,是无人系统的未来发展趋势。相较于传统无人系统,智能无人系统具备多种异构计算资源如中央处理器(central processing unit,CPU)、图形处理器(graphics processing unit,GPU)、可编程阵列(field programmable gate array,FPGA)和专用集成电路(application specific integrated circuit,ASIC)等,能够对智能计算任务充分发挥其架构优势,突破传统计算技术的限制,提供多样性的计算能力。

智能无人系统通常配备多种感知传感器,为智能无人系统提供全面、准确和实时的环境感知能力,从而实现更高效、更安全和更可靠的自主运动。随着智能无人系统感知传感器技术的不断发展,感知类传感器的数量、种类和精度不断增加。早期的智

① 国家重点研发计划(2022YFB4501600),中国科学院战略性先导科技专项(B 类)(XDB44000000)和中国科学院计算技术研究所创新课题(E261040)资助项目。

② 男,1990 年生,博士生;研究方向:计算机系统结构;联系人,E-mail:quanzhenyu314@126.com。

(收稿日期:2023-05-16)

能无人系统使用超声波雷达或摄像头来对周围的环境进行感知。如今,智能无人系统已经开始使用距离传感器、惯性传感器、红外热成像、全球定位导航系统等设备,并采取多传感器融合的方法,对周围的环境进行精准地分析。智能无人系统中的传感器可以收集到大量的数据,包括图像、位置、温度、压力、声音等信息。虽然多种高精度传感器能够有效提高智能无人系统的环境感知能力,但传感器数量越多、精度越高,其产生的原始数据量就越大,这些数据都需要传输、存储和实时处理。为了能够高效可靠地处理这些海量数据,智能无人系统不仅需要异构且强大的算力,还需要嵌入充分优化的数据处理框架才能充分发挥其性能优势。目前业界虽然出现了一些能够部署在智能无人系统上的数据处理框架,但这些框架主要用于处理巨量的无界数据流,并且这些框架与人工智能技术的结合程度较低,对智能无人系统的适配度不高,导致数据处理延迟无法满足智能计算任务的需求。

目前,各研究领域中面向特定计算平台的数据处理框架,其实现方法通常是将数据特征、硬件功能与科学的智能范式相结合,定制适用于各自研究领域的专用数据处理框架[2-3]。在设计数据处理框架时,智能范式的选择非常重要,这种范式需要具备很强的通用性和丰富的表达性,能够包含智能应用的主要特征。观察-判断-决策-行动(observe-orient-decide-act,OODA)模型是国内外著名的迭代反馈模型[4],该模型用观察、判断、决策和行动的循环过程来描述人类决策过程。在 OODA 模型中,OODA 的循环速度至关重要,它决定了个体或组织快速适应复杂环境的能力,循环速度越快,适应能力和竞争力就越强。OODA 模型最初用于战场决策,现已扩展到用于解释决策处理过程而得到了广泛应用。近年来,OODA 模型被用于指导面向群体智能无人系统的新型操作系统设计[5]。此外 OODA 模型被应用于无人机集群仿真建模框架的设计中[6],甚至还被运用到未来战斗机的发展路线图中[7]。OODA 模型的处理过程从观察开始,到行动结束,与环境存在极强交互性,智能无人系统的数据处理过程(信息收集、提炼、分析和实现控制)特征与 OODA 模型内涵匹配。因此,本文在设计面向智能无人系统的数据处理框架时,使用 OODA 模型来表征智能无人系统的数据处理过程,通过优化该 OODA 模型的循环速度来达到优化智能无人系统数据处理性能的目的。

本文基于 OODA 模型思想,设计了一套面向智能无人系统的流式数据处理框架——OODAFlow。OODAFlow 将 OODA 模型应用于智能无人系统的资源管理和智能计算任务的处理流程中,向下能够对智能无人系统中各类异构计算资源进行高效调度和管理,向上能够对复杂的智能计算任务负载进行处理,将复杂的智能计算任务在智能无人系统进行部署、调度和迁移,保证智能计算任务的高效处理以及异构计算资源的利用最大化。本文主要包含 3 方面工作:(1)提出了一种面向智能无人系统的流式数据处理框架 OODAFlow,基于 OODA 思想实现对智能无人系统异构资源的调度和智能计算任务的处理;(2)基于 OODAFlow 框架在智能无人系统上搭建了一套 OODA 任务处理系统,并提出了多种针对性的性能优化方法;(3)测试了 OODA 任务处理系统在无人机智能控制任务中的数据处理性能,测试结果表明,OODA 任务处理系统能够有效提高智能无人系统的数据处理速度,可满足智能计算任务对数据处理实时性的需求。

1 相关工作

1.1 智能无人系统

目前,无人系统的应用场景正往动态非结构化的方向发展,但传统的无人系统只针对特定的计算任务而设计,泛化能力较弱,只能够在特定限制条件下实现自主作业,难以适应新数据和任务的需求[8]。进入 21 世纪以来,人工智能技术的广泛应用为无人系统的发展提供了新的机遇。人工智能技术能够为无人系统提供复杂场景下的自主作业能力,提高无人系统在数据处理方面的能力,有效加速观察-判断-决策-行动环路速度。随着人工智能技术和无人系统的深入融合,智能无人系统应运而生。

智能无人系统是无人计算系统的一个重要分支,是一种通过深度学习技术来实现智能的无人系

统。智能无人系统具有环境感知和交互能力,能够适应多变的环境和需求,是无人系统的未来发展趋势。相较于传统无人系统,智能无人系统的特色在于具备专为人工智能技术设计的软硬件架构。在硬件方面,智能无人系统通常集成了 CPU 和不同硬件架构的人工智能芯片,是一种集成了多种异构计算资源的硬件系统。人工智能芯片已经成为提高智能无人系统计算效率的重要基础,这种芯片专为深度学习算法设计,能够有效提高智能无人系统处理深度学习算法的效率。在软件方面,智能无人系统通常支持不同种类的深度学习框架,支持不同类型的深度学习算法来为智能无人系统提供多种智能功能,还能集成一套编程环境,方便开发人员在智能无人系统上开发智能应用。目前,业界还出现了一种基于 OODA 模型的智能无人系统,即 OODA 计算系统[9],该系统采用了多加速器异构节点设计,根据 OODA 模型的特点,将整个计算系统分为主控分区、观察分区、判断分区、决策分区和行动分区,通过高性能的分区间互连实现松耦合通讯,并通过多种计算分区的协同加速实现最大化性能提升。

智能无人系统的优势在于智能化,即具有类似人类的思考推理能力。智能无人系统通过发挥其异构计算资源的性能优势,结合先进的深度学习算法,实现了环境感知和规划决策,能够应对多种难以预料的、非结构化的环境状况,更好地满足多样化的应用需求。智能无人系统主要部署在边缘端,更贴近数据源,可以就近对数据进行分析并作出反馈,减少了数据传输延迟。智能无人系统可以在本地高效完成深度学习算法的计算,无需再上传到云端服务器执行,具备更高的数据处理效率。智能无人平台由于具有较高的移动性,对环境条件、网路带宽等要求不高,因此可以被广泛部署,具有更广阔的发展前景和市场潜力。近年来,智能无人系统发展迅猛,出现了无人机、无人车、无人船等一系列产品,并且应用到制造业、交通等多个领域[10-11]。

1.2 数据处理框架

数据处理框架是智能无人系统提升数据处理效率的关键,目前业界已经出现了一些能够部署于智能无人系统的数据处理框架,这些框架主要分为 2 种类型,分别是批处理框架和流式处理框架。主流的批处理框架如 Apache Hadoop[12] 和 Apache Spark[13] 主要用于按批处理较大规模的数据,且需要数据预先存储在固定的存储位置中。然而在智能无人系统的应用场景中,数据通常是持续生成的,并且对数据处理的实时性有较高要求,因此批处理框架并不适合应用在智能无人系统中。

流式数据处理框架可以随时对进入系统的数据进行实时处理,更适合应用于智能无人系统。目前主流的流式数据处理框架有 Spark Streaming[14]、Apache Storm[15]、Apache Flink[16] 等,这些流式数据处理框架主要针对分布式集群系统设计,用于处理巨量的无界数据流、进行大数据计算,着重强调高性能计算[17]。但是这些框架能够处理的数据格式比较简单、通用,而智能无人系统中待处理数据的格式相对比较复杂。智能无人系统具备多种专为智能计算任务设计的异构计算资源,而主流的流式数据处理框架并不适合管理和调度这些计算资源,因此,主流的流式数据处理框架与智能无人系统的适配度不高。

目前,各研究领域中面向特定硬件系统的数据处理框架,其实现方法通常是将任务数据特征、硬件功能与科学的智能范式相结合,定制适用于各自研究领域的专用数据处理框架。本文将 OODA 智能范式与流式数据处理框架相结合,设计了一种面向智能无人系统的流式数据处理框架 OODAFlow。该框架能够充分利用智能无人系统的异构计算资源,将智能计算任务与 OODA 思想相融合,实现了 OODA 任务创建、任务调度、资源调度等功能,能够有效提高智能无人系统的数据处理性能。

2 OODAFlow 流式数据处理框架

智能无人系统的流式数据是指来自系统中各种传感器或设备的实时数据流。这些数据会不断地产生,并且有着不同的类型和来源,例如视觉摄像头、雷达、全球定位系统等。这些流式数据具有高速性和时效性,能够提供实时的信息和反馈。智能无人系统需要实时地将这些数据进行快速地传输、处理

和分析,并采取相应的决策和行动以保证系统能够正常完成任务目标。智能无人系统的流式数据处理框架是针对实时收集到的流式数据进行处理和分析的一套软件框架,这种框架是智能无人系统实现数据驱动实时决策的关键技术。合理的流式数据处理框架的设计和使用,可以帮助智能无人系统快速、精确地获取和分析流式数据,并快速做出智能决策,提高系统的效率和可靠性。

面向智能无人系统的流式数据处理框架通常需要结合系统的任务特征和硬件特征进行设计,并且最好有一种智能处理范式作为牵引。智能无人系统的特色在于自主性和智能化,具有自动适应外部环境的能力。而这种能力的形成,首先需要智能无人系统通过各种传感器观察周围环境,实时收集包括图像、声音、特定传感器数据等各种信息,对环境进行感知;其次需要智能无人系统通过深度学习等技术,在环境感知的基础上进行数据分析和推理,从而做出更加准确和高效的判断;此外还需要智能无人系统结合对于数据流的判断,根据情况调整任务策略和行动计划;最后需要智能无人系统采取行动来完成任务目标。智能无人系统的任务处理过程就是在不断观察、判断、决策和行动的过程中快速执行计算任务,完成任务目标。OODA 模型的目的是帮助个体或组织在快速变化和不确定性的环境中快速适应并制定有效的决策。循环速度是 OODA 模型的关键,循环速度越快,代表个体或组织具有更强的适应能力和竞争力。由于 OODA 模型所包含的观察、判断、决策和行动 4 个步骤与智能无人系统的任务处理流程非常相似,因此,本文在设计面向智能无人系统的数据处理框架时,使用 OODA 模型来表征智能无人系统的数据处理过程,将复杂的智能无人系统数据处理性能优化问题,转化为对 OODA 模型循环速度的优化问题。本文提出了一种面向智能无人系统的专用流式数据处理框架——OODAFlow。OODAFlow 将 OODA 思想与流式数据处理框架相结合,针对智能无人系统的硬件特征,采用自底向上的设计思路,逐层对任务处理流程进行抽象,通过在各业务层间设计连接接口,对各层结构进行相互连接和扩展,最终形成 OODAFlow 的框架结构。如图 1 所示,OODAFlow 流式数据处理框架由基础层、数据层、管理层和展现层组成。

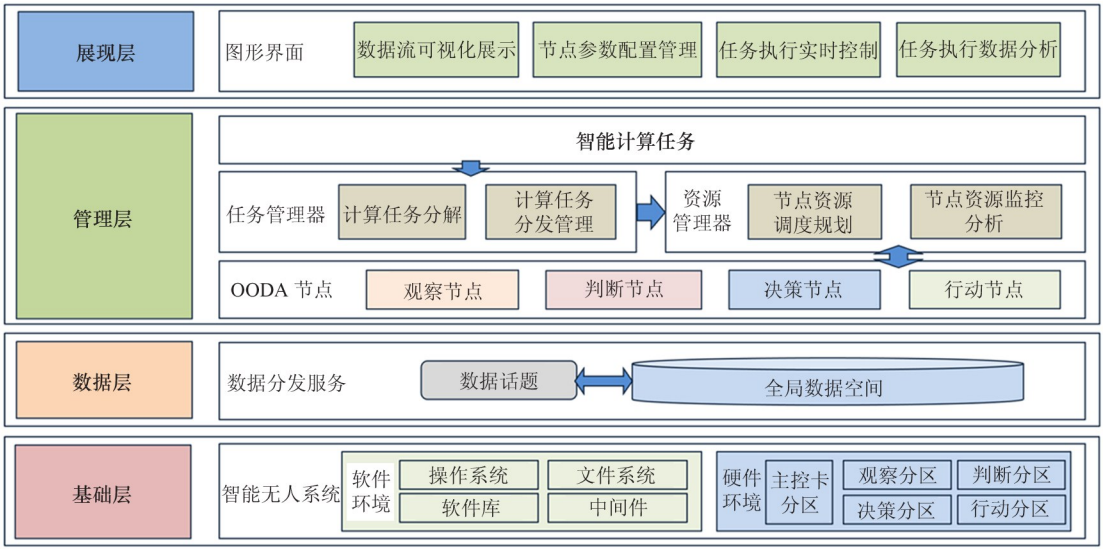

图 1　OODAFlow 框架层次结构

2.1　基础层

OODAFlow 流式数据处理框架主要面向智能无人系统设计,因此其基础层主要由智能无人系统的软硬件环境构成;其软件环境主要由操作系统、文件系统、各种软件库以及各类加速卡配套的中间件构成,用于支撑 OODAFlow 的资源管理和任务调度功能。

OODAFlow 将智能无人系统的硬件计算资源按

照主控卡分区、观察分区、判断分区、决策分区以及行动分区进行划分。主控卡分区所包含的计算设备用于实现高效异构资源管理和调度功能,是智能无人系统硬件资源的核心管理单元和调度“大脑”;观察分区的设备主要用于从外部设备中获取待处理的任务和数据,能够对获取到的数据进行预处理;判断分区主要由深度学习加速卡组成,能够实现目标检测、图像分类等深度学习算法的加速计算;决策分区包含的设备主要用于对判断节点输出的数据进行分析计算,形成决策内容;行动分区的设备主要用于同外部设备进行通信和控制,实现实时控制功能,是智能无人系统与外部设备之间的纽带。

2.2 数据层

OODAFlow 的数据层主要用于完成数据存储和信息通信任务。传统的计算系统通常采用树状结构,数据传输主要依赖中心节点,一旦中心业务节点崩溃,整个系统将失去数据通信和计算能力,节点间耦合性强、数据传输效率低。为满足智能计算任务对数据通信实时性和可靠性的需求,OODAFlow 需要避免中心节点失效后导致整个系统的瘫痪,同时需要保证数据传输的准确性、系统活跃程度以及安全性。高效的数据通信是实现智能计算任务处理流程的基础,OODAFlow 需要保证数据在各个计算节点中快速、高效、准确的传递,这就对框架的时延、可靠性和吞吐量等技术指标提出了较高的要求。

目前用于数据分发通信模型可以分为点对点、消息队列以及订阅发布 3 种模型。点对点模型即客户/服务通信模型,在处理任务时,客户端可以向一个或多个服务端请求服务,服务端可以处理来自一个或多个客户端的请求,执行完请求的计算任务后将计算结果返回给对应客户端。在这种通信模式中,客户端和服务端具有较高的时空耦合性,但灵活性、鲁棒性和扩展性较差。信息队列模型通过“队列”的数据结构来实现消息传递,生产者或消费者可以向一个或多个队列发送或接收消息。消息队列模型虽然解决了点对点通信方式中时空高耦合性的问题,但该模型无法实现异步消息通信,且可靠性较差。

在订阅发布通信模型中,消息的发布者将消息发送到“话题”中,然后,感兴趣接收这些消息的订阅者可以订阅这些“话题”,以便在消息可用时接收它们。该模型允许同时存在多个订阅者和发布者,发布者和订阅者之间的解耦使得它们可以独立工作,不会因为对方的变化而受到影响。订阅者可以选择订阅他们感兴趣的特定消息或事件,从而只接收他们关心的信息。由于发布者和订阅者是异步通信的,因此他们不必等待彼此的响应,从而提高了系统的整体性能。订阅发布模型通过使用代理来实现发布者和订阅者之间的解耦合,从而减少了直接交互所带来的复杂性。此外,该模型还可以轻松地添加新的订阅者和发布者,而无需对现有组件进行修改。相较于点对点和消息队列模型,订阅发布通信模型具有更多优势,并且更加贴合 OODA 计算系统的数据处理流程,因此 OODAFlow 数据层选择订阅发布通信模型来实现其数据通讯功能。

数据分发服务(data distribution service,DDS)是对象管理组织(object management group,OMG)制定的新一代分布式实时通信中间件技术规范[18]。数据分发服务基于订阅发布体系架构,强调以数据为中心,具有出色的解耦合特性以及高效实时的传输能力,并支持丰富多样的服务质量(quality of service,QoS)机制,目前已经被广泛应用于航空、国防、工业自动化等多个领域[19-21]。数据分发服务具有低延迟、高吞吐量、可控传输性能的优点,能够满足智能计算任务对实时性、安全性和灵活性的需求,因此 OODAFlow 的数据层基于数据分发服务来进行设计和实现。

如何快速高效地完成数据通信任务,是流式数据处理框架需要考虑的关键问题。数据分发服务采用以数据为中心的订阅发布模型(data-centric publish-subscribe,DCPS),如图 2 所示。数据分发服务定义了话题、发布者、订阅者、数据写入者以及数据读取者这 5 类通信实体[22]。DCPS 模型提出了全局数据空间(global data space,GDS)的概念,把系统中所有的本地数据都存储在全局数据空间中。在全局数据空间中,信息共享是以话题作为媒介,所有通信节点既可以是话题的订阅者、也可以是发布者,每一个节点都可以按照定义好的数据格式从全局数据空

间中订阅它感兴趣的数据或者发布它能提供的数据。在基于数据分发服务打造的 OODAFlow 数据层中,所有 OODA 节点的数据共享都是通过话题机制来驱动,数据消息在节点间以点对点的形式完成高速通信共享。

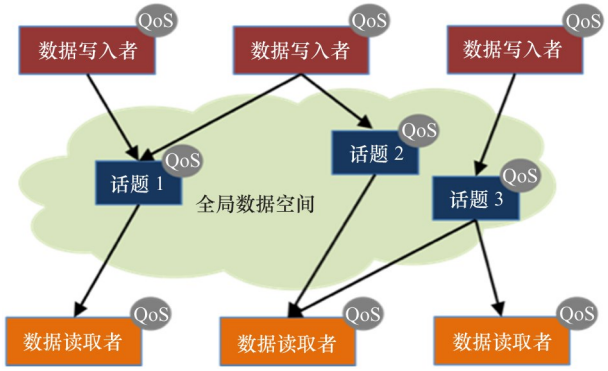

图 2　数据分发服务结构

基于数据分发服务来实现的数据层更具优势。由于数据层中并不需要维护传统计算系统中的中心业务节点,每个节点只需要关注自己感兴趣的话题,而无需知道数据发布者是谁。节点间只有在话题订阅成功时才建立通信,节点间通过全局数据空间中的话题进行关联,能够实现一对多的连接,有效地提高了数据通信效率。当话题订阅成功时,话题数据由发布节点以点对点的方式直接发送给订阅节点,不存在任何中间节点,通信成本开销更小。节点间可以通过 QoS 策略选择最符合双方节点情况的传输方法,有效增加了数据通信吞吐量,增强了数据通信实时性。此外,在系统运行过程中,系统可根据自身需要动态增加或删除通信节点,来为系统扩充或减少计算资源;也可以随时添加或删除数据话题,方便对系统功能进行调整。

2.3　管理层

管理层主要负责完成智能计算任务管理、OODA 任务分发和资源调度工作,主要由任务管理器和资源管理器构成。任务管理器负责智能计算任务全生命周期的管理,是衔接 OODAFlow 上下软硬件层的关键。任务管理器的功能是将智能计算任务抽象分解为观察、判断、决策和行动 4 种子任务,并将这些子任务的 OODA 节点配置信息和数据打包分发给资源管理器。资源管理器负责将智能无人系统的计算资源划分为观察、判断、决策和行动 4 个节点进行管理,并进一步将各节点分解为更细粒度的功能算子来进行调度,以实现所有智能计算任务的并行调度和动态资源规划;同时还具备 OODA 节点运行状态监控和分析的能力。

2.4　展现层

展现层主要用于向用户提供数据图形化界面以及人机交互接口。展现层基于 HTML、Java Script 开发,并在底层实现代码中接入 DDS 服务,通过订阅和接收各 OODA 节点的话题数据,能够实时显示智能计算任务在各 OODA 节点的运行状态。用户可以通过与展现层进行交互,来向 OODA 节点的控制接口话题发送相应的指令,对各节点进行配置和控制。

3　OODAFlow 管理层实现

OODAFlow 管理层主要负责完成智能计算任务的解析和分发,能够实现智能无人系统计算资源的管理和调度,是连接智能计算任务和异构计算资源的关键一环。管理层的设计思路是将智能计算任务分解为一系列的 O-O-D-A 子任务,将智能无人系统的计算资源抽象为 O-O-D-A 4 个节点进行管理和调度。并针对智能计算任务的计算特征对任务处理流程进行优化,以加速数据处理速度。

OODAFlow 管理层的结构如图 3 所示,管理层通过各个模块之间的紧密配合来完成智能计算任务的处理流程。首先,管理层将智能无人系统中的各类异构计算资源在软件层面上封装为观察、判断、决策以及行动节点进行管理和调度。任务管理器在接收到智能计算任务后,其计算任务分解模块将对任务信息进行分析和分解,并将任务信息打包为 OODA 任务实例。之后,通过任务分发管理模块将 OODA 任务实例发送给资源管理器。资源管理器在接收到 OODA 任务实例后,其资源调度模块将按照任务内容对各 OODA 计算节点进行调度,从而完成该智能计算任务的计算过程。

3.1　OODA 节点划分

OODAFlow 的管理层根据智能无人系统中各类异构计算资源的主要功能,在软件层面上将这些计

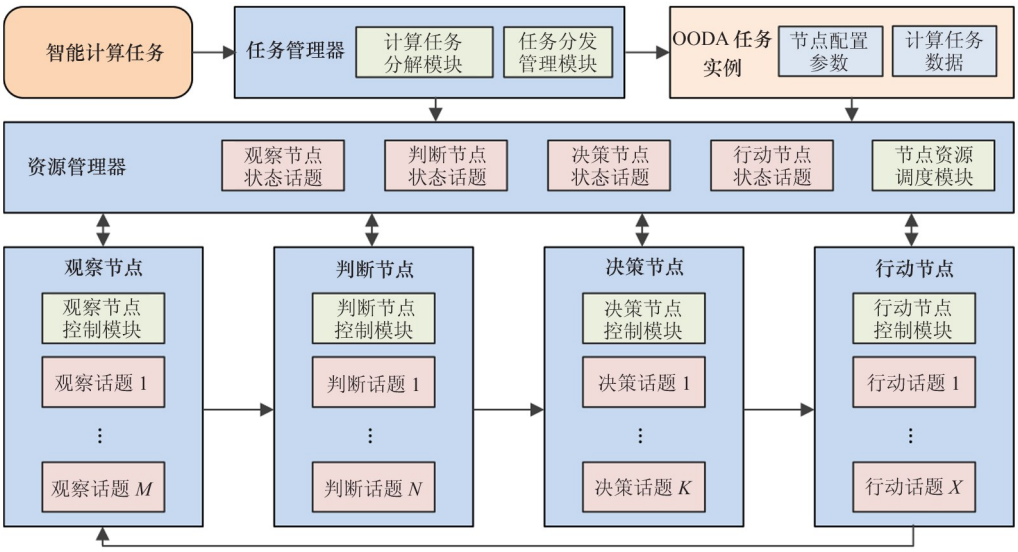

图 3　OODAFlow 管理层结构

算资源封装为观察、判断、决策以及行动节点,将各计算资源节点按照观察-判断-决策-行动的流程进行管理和整合,形成完整的体系回路。OODAFlow 对各 OODA 节点的具体划分思路如下。

观察节点:智能无人系统中具备从外部设备中获取智能计算任务信息和数据的硬件设备,例如图传处理卡等,这类设备能够实现 OODA 模型的观察环节,因此将这类设备在软件层面上封装为观察节点进行管理。

判断节点:智能无人系统中的深度学习加速卡能够有效加速深度学习算法的计算效率,可以通过深度学习算法对输入数据进行分析,提炼出其中的关键信息。该功能与 OODA 模型的判断环节相符,因此将深度学习加速卡封装为判断节点进行调用和管理。

决策节点:决策环节主要是通过使用决策算法对判断结果进行分析,形成决策信息,因此管理层将能够完成决策算法计算的硬件设备封装为决策节点进行调用。

行动节点:行动环节主要是根据决策信息计算得出行动方案,并将行动方案输出到指定的外部设备中,实现行动功能,管理层将具备上述功能的设备封装为行动节点进行管理。

3.2　任务管理器

任务管理器主要用于对智能计算任务进行分析,按照观察-判断-决策-行动流程对任务进行分解,将任务信息打包发送给资源管理器。任务管理器主要由计算任务分解模块和任务分发管理模块构成。

计算任务分解模块负责将智能计算任务按照 OODA 模型进行解析、分解,并重新封装为具有固定数据格式的 OODA 任务实例。针对不同类型的智能计算任务,任务分解模块首先对智能计算任务进行解析,获取任务信息和数据;之后按照 OODA 模型对智能计算任务的计算流程进行分解,生成相应的 OODA 节点配置参数。OODA 节点配置参数定义了一套统一且固定的数据结构,包含了对各 OODA 节点功能的配置信息,用于驱动各 OODA 节点实现不同类型计算任务的处理流程。配置参数中还定义了计算任务的任务优先级,当系统中出现资源竞争的情况时,资源管理器能够按照计算任务的优先级信息为重点计算任务优先分配计算资源。

任务分发管理模块的处理流程由 OODA 任务实例驱动,当接收到由计算任务分解模块生成的 OODA 任务实例后,立即将该数据发送给资源管理器进行后续处理。OODA 任务管理器通过与资源管理器进行通信,保证每个 OODA 任务实例都能够完整、正确地发送到资源管理器中,为后续 OODA 计算流程做准备,保证智能计算任务处理过程的正确性。

3.3　资源管理器

资源管理器主要用于对智能无人系统的异构计算资源进行管理和调度，通过节点资源调度模块和各 OODA 节点的节点控制模块来实现资源管理功能，控制各 OODA 节点完成智能计算任务的数据处理流程。

OODA 节点间的数据处理流程通过话题机制驱动。各 OODA 计算节点在完成计算任务后，将计算中间结果数据发布到该节点的对应话题中。下一计算节点通过订阅该节点所发布的相应话题来获取中间计算结果数据，从而开始进行下一步任务计算流程。同一个计算节点中可以同时维护多个话题，也可以同时向多个话题发布数据消息，从而在同一节点中支持多任务并行处理，加速任务处理速度。各 OODA 节点中的节点控制模块用于接收来自节点资源调度模块的控制信息，节点控制模块能够对该节点的话题参数进行配置，通过修改该节点订阅和发布话题的名称，来将该节点与前后计算节点进行连接，以针对不同类型的计算任务实现不同的计算流程。

资源管理器一旦接收到由任务管理器发送的 OODA 任务实例，将立刻通过节点资源调度模块控制所有相关 OODA 节点开始该任务的计算流程。资源调度模块能够对 OODA 任务实例进行解析，根据其中的节点配置参数，对各 OODA 节点的控制模块进行配置和控制，操控所有 OODA 节点按照观察-判断-决策-行动的顺序逐步完成该智能计算任务。资源管理器中维护了所有 OODA 节点的状态话题，各节点通过向自己节点的状态话题发布消息来上报自己的运行状态。资源管理器通过分析各 OODA 节点的运行状态来判断智能计算任务的执行情况，确定每个节点的空闲资源情况，并以此来优化资源调度策略。

每个 OODA 节点都能够并行处理多个计算任务，当各节点出现计算资源紧张的情况时，资源管理器将结合各 OODA 节点的运行状态，按照各计算任务的优先级对计算资源进行动态分配，保证重要任务优先完成处理。资源管理器根据各智能计算任务的计算特征，结合各节点的运行状态，对 OODA 节点的计算资源进行动态规划，保证 OODA 计算资源能够被充分利用。

3.4　OODAFlow 管理层工作流程

OODAFlow 管理层的工作流程如图 4 所示。任务管理器的任务解析模块能够接收待处理的智能计算任务，对智能计算任务进行分析后，模块基于 OODA 思想和各 OODA 节点的具体功能对智能计算任务进行分解，并生成各 OODA 节点的配置参数。任务解析模块将任务数据和节点配置参数打包为 OODA 任务实例，之后由任务分发模块将 OODA 任务实例发送给资源管理器，以启动任务的处理流程。

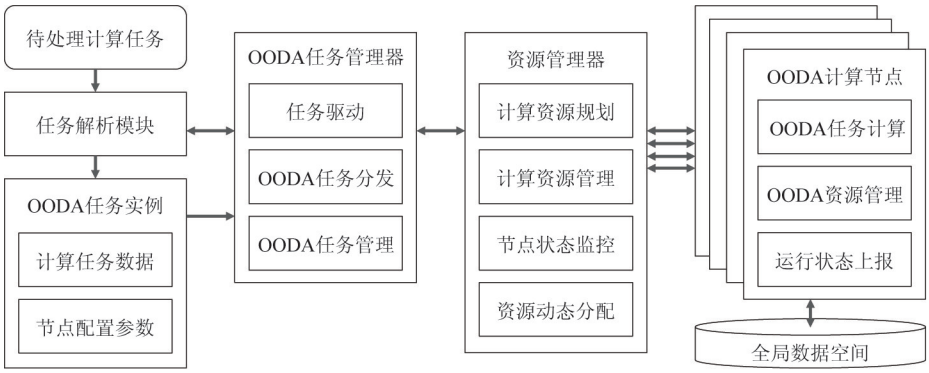

图 4　OODAFlow 管理层的工作流程

资源管理器接收到 OODA 任务实例后，立即调度各 OODA 节点进行任务处理流程。OODA 节点中所有数据都存储在全局数据空间之上，各 OODA 节点基于话题机制在全局数据空间之中实现数据共享。智能计算任务在各 OODA 节点中按照观察-判断-决策-行动的计算流程进行处理，资源管理器通过监控各 OODA 节点的运行状态，对任务的处理流程进行控制。针对高优先级的计算任务，资源管理

器将优先分配计算资源。

4 OODAFlow 的数据流处理模型

OODAFlow 框架的数据处理流程包括观察、判断、决策和行动 4 个阶段,数据流处理过程如图 5 所示。

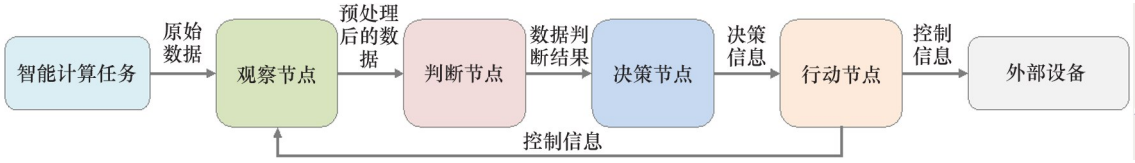

图 5 OODAFlow 的数据处理模型

第 1 阶段为观察。观察节点与外部设备相连接,如无人机、图像传感器等,观察节点能够从 OODA 任务实例中获取数据来源信息,根据该信息从指定的外部设备中获取包含周围环境信息的数据。在完成周围环境情报的收集工作后,观察节点将按照智能计算任务的类型,对原始数据进行预处理。最后将处理后的环境信息数据发送到观察节点的指定话题。

第 2 阶段为判断。判断节点主要通过深度学习加速卡运行深度学习算法来对周围环境信息进行处理。判断节点根据 OODA 任务实例的参数配置信息,完成对深度学习加速卡的参数配置,可配置的参数包括使用的深度学习算法参数、投入计算的深度学习加速卡数量、每张深度学习加速卡的并行度等。通过使用深度学习算法对输入数据进行计算后,将计算结果发送到判断节点的指定话题。

第 3 阶段为决策。决策阶段的主要任务就是制定明确的任务决策方案、下达决策计划,决策节点按照 OODA 任务实例的配置信息,根据不同任务使用特定的决策算法对判断节点的计算结果进行分析计算,最终形成决策信息,之后将该信息发送到决策节点的指定话题。

第 4 阶段为行动。行动阶段主要针对决策方案采取相应的控制手段,控制无人机等外部设备开展动作行为。这一阶段中,行动节点中的控制加速处理卡将对输入的决策信息数据进行处理,得出面向外部设备的具体控制数据,通过通信手段将控制数据发送控制信息到外部设备,从而实现控制功能。此外,行动节点将控制数据同步发送到行动节点的指定话题,观察节点通过订阅该话题来获取行动信息,作为其获取输入数据时的参考信息。

5 OODA 任务处理系统

本文基于 OODAFlow 框架在 OODA 计算系统上构建了一套用于处理智能计算任务的 OODA 任务处理系统。在 OODA 任务处理系统中,OODA 节点是完成任务处理的关键,针对主流智能计算任务的计算特征,本文为 OODA 4 个节点分别设计和实现了不同的功能算子,通过 OODA 节点中各功能算子间的组合,能够完成对不同类型智能计算任务的处理。

5.1 OODA 计算系统

OODA 计算系统[9]是一种基于 OODA 模型的智能无人系统,如图 6 所示,OODA 计算系统由“1 个机箱 +5 个功能分区”的硬件结构及配套的软件环境组成,其特色在于搭载了用于加速深度学习算

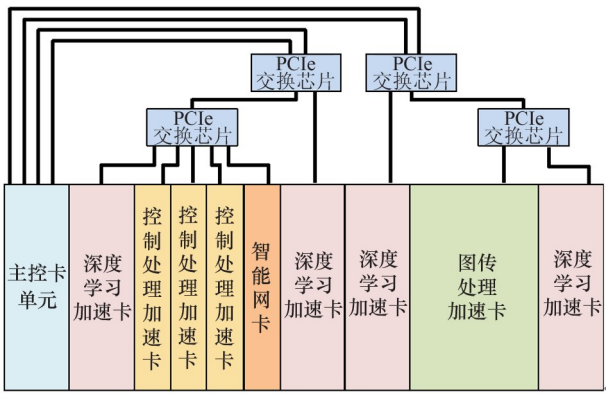

图 6 OODA 计算系统硬件连接逻辑

法处理速度的深度学习加速卡。OODA 计算系统的软件环境主要由操作系统、文件系统、各种软件库以及各类加速卡配套的中间件构成。OODA 计算系统的硬件结构采用了多加速器异构节点设计,根据 OODA 模型的特点,将整个计算系统分为主控卡分区、观察分区、判定分区、决策分区和行动分区。所有功能分区中的加速卡均采用面向仪器系统的外设高速互联扩展总线协议(peripheral component interconnect extensions instrumentation express, PXIe)打造,加速卡间通过计算机外设高速互联总线协议(peripheral component interconnect express, PCIe)交换芯片进行数据通信。

主控卡分区的主控卡基于飞腾 FT1500A/16 研制,整板峰值功耗约 55 W。该分区还配置有 16 GB 的国产紫光板载内存芯片,并采用了板载 64 GB 固态存储芯片。由于主控分区中的处理器还可以用于完成决策算法的计算过程,因此该主控卡在软件层面上还被封装为决策分区进行管理。

观察分区的图传处理加速卡基于英特尔酷睿 i7-6820 处理器打造,具有实时处理图传、数传信号数据处理以及构建无人机环境态势图等功能,是 OODA 计算系统内的图传数据预处理单元,也是从外部设备中获取数据的窗口。

判断分区共包含 4 块深度学习加速卡,深度学习加速卡基于寒武纪 MLU100 深度学习加速模块研制,配备有 16 GB 板载内存,内存带宽能达到 102.4 GB·s^{-1},板卡峰值功耗约 80 W,采用被动散热方式,支持双向风道进风。

行动分区由控制加速处理卡与智能网卡构成。控制加速处理卡由赛灵思 KintexUltrascale 040 FPGA、双倍数据速率存储器、PXIe 连接器等部分组成,配置了运动控制类算法、PCIe 直接内存访问软件包、PCIe Gen 2.0 x8 软件包,能够完成姿态和运动控制加速功能。智能网卡基于赛灵思 Kintex 7325 FPGA 打造,采用 PXIe 制式,主要实现的功能是与外部设备之间进行高带宽低延时的通信。

5.2 OODA 节点功能算子实现

OODA 任务处理系统将 OODA 计算系统中的观察、判断、决策以及行动分区在软件层面上封装为 OODA 节点进行管理。由于各 OODA 节点都能够实现多种计算功能,为了实现更细粒度的计算资源调度,OODA 任务处理系统将各个 OODA 节点进一步分解为更具体、更易于调度的功能算子,并对功能算子进行封装,提供相应的算子调用接口,通过组合每个 OODA 节点的不同功能算子来实现智能计算任务的数据处理流程。此外,细粒度的功能算子机制也为后续的性能优化提供了可能性。

观察节点中的算子功能主要包括:(1)从外部设备获取图像:观察节点使用 TCP/IP 协议与外部设备进行通信,以获取任务信息和数据,并将接收到的数据发送到指定的话题。(2)图像预处理:对输入的图像进行剪裁、缩放、压缩、填充、归一化等一系列预处理操作过程。

判断节点中的算子功能主要包括:(1)目标检测算法计算:使用 YOLO(you only look once)[23]、快速区域卷积神经网络(faster region-based convolutional neural network, Faster R-CNN)[24]等算法对输入的图像数据进行处理,找出图像中的指定目标,确定目标的位置、类别,并将识别到的目标信息发送到指定的话题。(2)图像分类算法计算:使用 ResNet[25]、VGG(visual geometry group)[26]等算法对输入的图像数据进行计算,确定图像所属的类别,并将分类结果发送到指定的话题。

决策节点中的算子功能主要包括:(1)A * 路径规划算法计算:使用 A * 路径规划算法对目标检测的结果数据进行处理,根据目标检测算法分析得到的目标位置,通过计算得出路径轨迹信息,并将路径规划信息发送到指定的话题。(2)查找决策表:针对判断节点的目标检测或图像分类算法的计算结果,查找本地的决策表文件,找出相应的决策信息,并将这一决策信息发送到指定的话题。

行动节点中的算子功能主要包括:(1)加速计算控制信息:针对决策节点发送的决策数据,使用控制计算加速卡进行处理,计算得出针对无人机或外部设备的具体控制信息。(2)控制智能网卡发送控制信息:将具体行动信息通过智能网卡发送到指定的硬件,完成对无人机或外部设备的控制任务。

6 OODA 任务处理系统性能优化方法介绍

OODA 任务处理系统将 OODA 计算系统的计算

资源划分为 O-O-D-A 节点进行管理,并将各节点能够实现的功能抽象为更细粒度的功能算子,这就为 OODA 任务处理系统提供了性能优化的可能性。本文提出了 3 项性能优化方法来优化智能无人系统的数据处理效率,分别是图像预处理过程优化方法、流水线优化方法以及判断节点并行加速优化方法。由于不同深度学习框架在判断节点上的性能表现各不相同,本文通过测试和对比 3 种深度学习框架在判断节点上的性能表现,确定了最适合判断节点的深度学习框架。

6.1 图像预处理过程优化方法

在某些智能计算任务中,智能无人系统从外部获取到的数据都是未经过任何处理的原始数据,这些数据虽然包含更多的信息量,但需要更多的算力和时间来对数据进行预处理。例如由观察节点从无人机端获取到原始图像,由于原始图像分辨率较高,在处理每帧图像时都需要经过行剪裁、缩放、压缩、填充、归一化等一系列预处理操作,因此预处理过程所需要的计算资源就更多。

图像预处理算法默认是由判断节点进行处理,而判断节点还负责使用深度学习算法对图像进行处理。判断节点是 OODA 节点中最为关键的一环,虽然判断节点中的深度学习加速卡具备加速深度学习算法处理速度的能力,但其并不具备加速图像预处理算法的能力。因此,当系统处理高分辨的图片时,就会占用判断节点过多的计算资源来进行图片预处理计算,从而降低其处理深度学习算法的计算效率,导致整个 OODA 任务处理系统的数据处理速度变慢。

针对这一问题,图像预处理过程优化方法的解决思路是将原本由判断节点负责处理的图像预处理过程进行拆分;将其中耗时较长的部分图像预处理算法,如剪裁、缩放、压缩等转移到观察节点的图传处理卡中完成,而部分基础图像预处理算法依然由判断节点进行处理。由于图传处理卡的主控具有较强的图形处理能力,可有效加速图像预处理速度,通过在图传处理卡中完成大部分图片预处理过程,再将处理后的图片传送到判断节点的深度学习加速卡中进行计算,这样能够有效加速判断节点的计算效率,进而缩短每次 OODA 循环所需要的时间。

6.2 流水线优化方法

OODA 任务处理系统将智能计算任务分解为观察、判断、决策和行动 4 种子任务进行处理。系统默认的处理流程是各 OODA 节点按照串行的方式顺序完成每个智能计算任务的计算流程。当采用这种处理逻辑时,某 OODA 节点在执行子任务时,其余 3 个节点都处于空闲状态,这就造成了计算资源的浪费,降低了数据处理效率。

针对这一问题,本文的解决思路是引入流水线的思想来处理智能计算任务。任务流水线优化方法是一种将数据处理任务分解为多个阶段,形成一个有序的处理流程以提高处理效率的方法。任务流水线优化方法包括并行计算、流水线拆分与调度、管理等技术,可以最大限度地发掘硬件的性能潜力,提高系统的吞吐量和响应速度。任务流水线的各个阶段可以并行处理不同的任务,从而充分利用系统资源,提高并行性,加速任务执行速度。任务流水线可以同时处理多个任务,且每个任务只需在其所在的阶段中等待,因此可以降低任务的整体等待时间和延迟。任务流水线优化方法可以方便地实现模块化设计,对于任务的增加或修改可以很容易地进行扩展和调整。任务流水线优化方法可以根据任务的特点和要求进行灵活配置,例如可以通过增加阶段数、调整任务分配策略等方式来实现更高的性能和效率。

当 OODA 任务处理系统引入了任务流水线优化方法之后,OODA 任务处理系统将智能计算任务拆分为观察(observe)、判断(orient)、决策(decide)和行动(act)4 种子任务,并在不同的处理阶段同时执行,从而减少了任务之间的等待时间,提高了整体运 行效率。如图7所示,在$T2$时间段,判断节点在

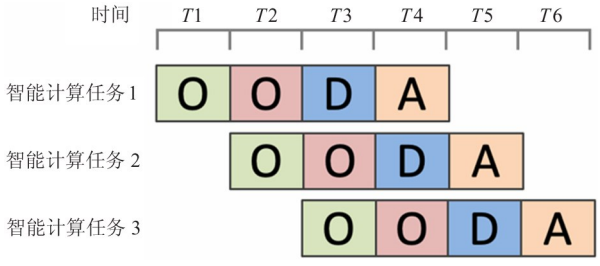

图 7 OODA 任务流水线优化方法原理

处理智能计算任务 1 的判断部分计算任务时，此时空闲的观察节点可以继续处理任务 2 观察部分的计算任务。以此类推，当所有的计算节点都采用这样的流水线处理逻辑时，就能够在 OODA 任务处理系统中实现多任务并行处理功能，从而提升整个系统的数据吞吐量，提高系统计算资源的利用率。

6.3　判断节点并行加速优化方法

判断节点由 4 块深度学习加速卡组成，当判断节点使用深度学习算法处理图像数据时，如图 8 所示，深度学习加速卡的默认计算逻辑是同时使用多张加速卡来处理每单位的数据。在这种逻辑下，判断节点只有在完成当前数据的处理过程后，才能够继续处理下一单位的数据。这样的计算逻辑虽然可以提高每单位数据的计算速度，但处理每单位数据所需的计算量远达不到多张深度学习加速卡的峰值计算性能，这就造成了判断节点硬件计算资源的浪费，导致判断节点数据吞吐能力的降低以及计算资源利用率的降低，严重影响了智能无人系统的 OODA 任务处理效率。

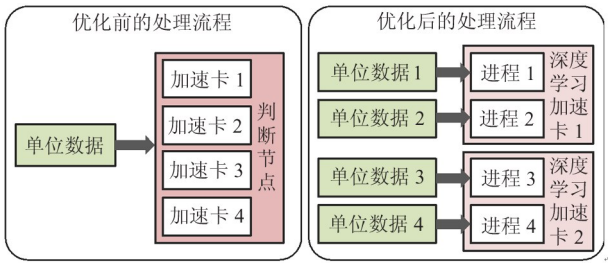

图 8　OODA 中第 2 个“O”(判断节点)的并行加速优化方法

针对这一问题，本文的解决思路是通过使用并行加速优化方法，改变深度学习加速卡的数据处理逻辑，让判断节点中的每块深度学习加速卡都能够以并行处理的方式同时处理多个数据。其实现方法是通过使用多个进程将每块深度学习加速卡在软件层面上抽象为多个虚拟计算节点，每个虚拟计算节点通过“话题”机制接入到全局数据空间之中。并且每个虚拟计算节点都维护一个判断话题，负责处理 1 单位的数据。通过资源管理器的调度，让每块深度学习加速卡通过自身抽象出的多个虚拟计算节点实现并行处理多单位数据的计算能力，有效提升判断节点的计算并发度、吞吐性能。此外，判断节点并行加速优化方法使判断节点具备了可扩展性，可以通过增加深度学习加速卡的数量或虚拟计算节点的数量，满足更高的计算要求。相较于传统的数据处理逻辑，判断节点并行加速优化方法能够有效提升判断节点的计算效率，进一步缩短 OODA 循环的处理时间。

6.4　确定最适合判断节点的深度学习框架

在 OODA 任务处理系统的 OODA 节点中，判断节点是计算量最大的节点，也是数据处理时间最长的节点。判断节点主要用于加速深度学习算法的计算过程，因此如何让判断节点更快地执行深度学习算法，成为了提高其数据处理速度的关键。影响深度学习算法处理速度的关键因素除了硬件本身的计算性能之外，另一因素就是深度学习框架的性能。目前业界已经出现了多种可以部署在智能无人系统中的深度学习框架，例如由伯克利于 2013 年发布的开源框架 Caffe[27]、谷歌于 2016 年发布的 TensorFlow[28]、亚马逊于 2017 年发布的 MXNet[29] 等。由于不同的深度学习框架在同一计算设备上的性能表现可能会存在差异，为了找出在判断节点的深度学习加速卡上性能表现最好的深度学习框架，本文选择了 3 种主流且兼容寒武纪 MLU100 深度学习加速卡的深度学习框架即 Caffe、TensorFlow 以及 MXNet 进行性能测试。

智能无人系统主要部署在边缘端，由于边缘端的计算任务更多的是执行神经网络推理任务，较少涉及神经网络训练任务，因此，本文对深度学习加速卡在 3 种神经网络框架上的推理性能进行测试。对于边缘端智能计算任务来说，数据处理速度和数据处理准确率尤为重要，因此，本文在一块深度学习加速卡上测试了 3 种深度学习框架的数据推理速度和准确率表现。测试中使用了 7 种主流的图像分类卷积神经网络模型，测试所使用的模型均采用 Float 16 数据格式。

深度学习加速卡在 3 种深度学习框架下的处理速度和准确率表现如图 9、10 所示。在处理速度的测试中，本文使用了每秒处理图像帧数(frame per second，FPS)作为评价指标。从图 9 中可以看出，3 种框架在不同神经网络模型下的处理速度各不相

同,其中大多数神经网络模型在 MXNet 下的处理速度都比较快,但 VGG 和 ResNet 系列神经网络模型在 MXNet 下的处理速度明显不如 Caffe 和 TensorFlow 快。在准确率的测试中使用了图像分类领域较为常用的前 5 准确率作为评价指标。从图 10 中可以看出,大多数神经网络模型在 MXNet 下都具有更高的准确率。虽然 VGG 和 ResNet 系列神经网络模型在 MXNet 下的处理速度最慢,但是 VGG 和 ResNet 系列神经网络模型在 MXNet 下的准确率最高。从综合性能测试结果可以发现,MXNet 在深度学习加速卡上处理大部分神经网络模型时都具有更快的处理速度和更高的处理准确率,因此判断节点选择 MXNet 作为处理深度学习算法的计算框架。

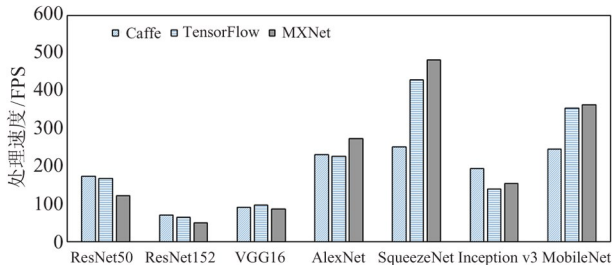

图 9　深度学习加速卡在 3 种深度学习框架下的处理速度表现

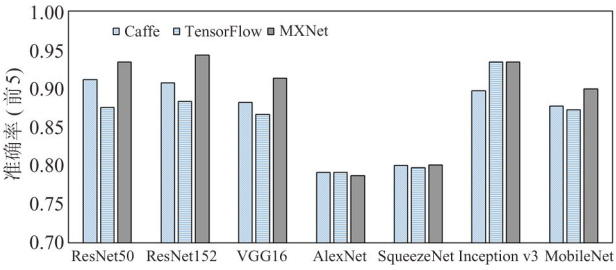

图 10　深度学习加速卡在 3 种深度学习框架下的准确率表现

7　OODA 任务处理系统应用效果分析

7.1　无人机智能控制仿真实验

本文通过无人机智能控制仿真实验来测试 OODA 任务处理系统的实际数据处理性能。无人机智能控制任务通过对无人机捕捉到的图像数据进行实时分析,识别出实验场地中蓝色机器狗的位置,根据目标位置生成决策信息,最终通过向无人机发送控制指令,控制无人机移动到目标位置。

无人机智能控制任务的计算流程可以按照观察、判断、决策和行动 4 个阶段进行划分。

观察阶段:对无人机的当前状态和环境进行观察,包括对周围环境的感知以及环境信息的收集和传输。

判断阶段:对观察到的信息进行分析和整合,完成对环境中机器狗位置和颜色的识别,确定下一步操作的决策依据。

决策阶段:根据前 2 个阶段的结果,进行智能决策。使用路径规划算法对目标识别结果进行处理,计算得出无人机移动到蓝色机器狗位置的飞行路径。

行动阶段:将决策的结果转化为实际的行动指令。对决策信息进行处理,输出面向无人机的具体控制数据,最终通过通信手段将控制数据发送到无人机,控制无人机按照决策完成任务。

无人机智能控制仿真实验在户外场地中进行,实验场地中摆放着 2 只带有红色和蓝色标识的机器狗。为了实现对机器狗颜色和位置的识别,本文构建一套机器狗数据集,用于对目标识别模型进行训练,最后通过训练后的目标识别模型来对场地中的机器狗进行识别。机器狗数据集的数据来自无人机实际拍摄的视频,共包含了 4 000 帧无人机高空俯拍视角下的机器狗图像,这些机器狗的机身上分别带有红色或蓝色的标识。机器狗数据集采用人工手动标注的方法对每个图像进行分类标签的标注,标注好的数据集按照 8∶1∶1 的比例划分为训练集、验证集和测试集。本实验采用 YOLO v3 模型来实现目标识别任务,并使用 MXNet 作为深度学习框架,YOLO v3 的模型大小为 61.93 MB,模型包含的计算操作数量为 27.38×10^9。在使用机器狗数据集对 YOLO v3 进行充分训练后,该模型的平均精度均值(mean average precision,mAP)为 0.95,模型精度可以满足任务需求。其识别效果如图 11 所示,训练后的模型用长方形的识别框标记出了带有蓝色标识的机器狗,实现了对机器狗颜色和位置的识别。

在性能定义上,本文选择 OODA 循环平均处理延迟作为评价智能无人系统数据处理性能的指标。OODA 循环处理延迟是指智能无人系统完成无人机智能控制任务中的观察-判断-决策-行动计算流程所

需的时间。本文统计并计算了一个智能无人系统在处理 1 000 帧无人机图像后的 OODA 循环平均处理延迟,并以该数值作为评价其数据处理性能的指标。为了测试 OODA 任务处理系统的性能优化效果,本文还采用了另一套具有相同配置的 OODA 计算系统作为对比系统,该对比系统采用常规的方法来完成无人机智能控制任务的 OODA 计算流程。

图 11　无人机图像目标识别结果

7.2　性能测试结果

本文测试了 OODA 任务处理系统在无人机智能控制仿真实验中的性能表现,还测试了 3 种性能优化方法对智能无人系统的优化效果,这 3 种性能优化方法分别是:方法 1 图像预处理过程优化方法、方法 2 流水线优化方法以及方法 3 判断节点并行加速优化方法。OODA 任务处理系统的性能测试结果如图 12 所示,图中展示了未搭载 OODA 任务处理系统的同配置对比系统的数据处理性能,以及采用了 OODA 任务处理系统及其 3 种性能优化方法后的 OODA 计算系统的性能表现。

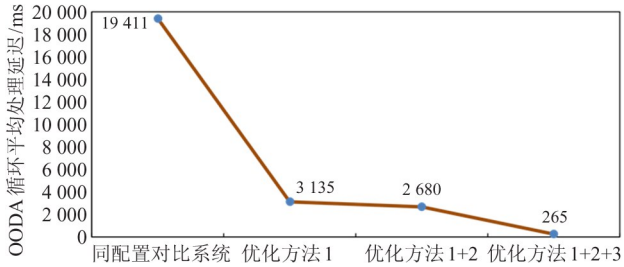

图 12　OODA 任务处理系统的性能表现

当不使用 OODA 任务处理系统来完成无人机智能控制任务时,由于常规的处理方法无法充分调度和利用智能无人系统中的计算资源,因此,同等配置的对比系统需要花费 19.410 s 才能完成一帧图像数据的处理流程,处理延迟较高,无法满足无人机智能控制任务对于控制实时性的需求。

当使用 OODA 任务处理系统来完成无人机智能控制任务时,在仅使用方法 1 图像预处理优化方法时,性能就得到明显改善,OODA 循环平均处理延迟降低到了 3.130 s。可见图像预处理流程严重影响了判断节点中深度学习加速卡的性能表现,而判断节点的延迟会直接影响整个 OODA 计算流程的总延迟。当采用图像预处理优化方法将部分计算量较大的图像预处理算法转移到观察节点中计算后,虽然观察节点的平均处理延迟从 0.077 s 增加到 0.884 s,但判断节点的平均处理延迟从原来的 19.330 s 降低到 2.250 s,从而显著降低了 OODA 任务处理系统的处理延迟。

在继续使用方法 2 流水线优化方法后,OODA 任务处理系统的性能得到了进一步提升,OODA 循环平均处理延迟降低到 2.683 s。在前 2 种优化方法的基础上继续使用方法 3 判断节点并行加速优化方法后,OODA 循环平均处理延迟显著降低。当使用 4 卡 4 进程时,即判断节点中的 4 张深度学习加速卡分别被 4 个进程接管,每张加速卡负责处理 1 张图像,OODA 循环平均处理延迟为 0.486 s。当使用 4 卡 8 进程时,即每张深度学习加速卡分别被 2 个进程接管,可以同时处理 2 张图片,拥有 4 张深度学习加速卡的判断节点能够同时并行处理 8 张图像,这时系统的 OODA 循环平均处理延迟降低到 0.265 s,系统的数据处理性能得到了极大的提高。

从上述性能测试结果中不难看出,相较于同配置的对比系统,在部署了 OODA 任务处理系统后,智能无人系统的数据处理性能提升了 73 倍。可见基于 OODAFlow 搭建的 OODA 任务处理系统能够显著提升智能无人系统的数据处理效率。

8　结　论

本文介绍了面向智能无人系统的流式数据处理框架 OODAFlow,该框架将 OODA 思想应用于智能计算任务的处理和计算资源管理的流程中,能够对智能无人系统中各类异构计算资源进行高效调度和

管理。本文基于 OODAFlow 框架在 OODA 计算系统上搭建了一套 OODA 任务处理系统,进一步验证了 OODAFlow 的可行性,并提出了多种具有针对性的性能优化方法,提高了系统的数据吞吐性能和资源利用率。本文利用无人机智能控制任务测试了 OODA 任务处理系统的数据处理性能。性能测试结果表明,相较于同配置的对比系统,OODA 任务处理系统的数据处理性能提升了 73 倍,充分发挥了智能无人系统的性能优势,提升了智能无人系统的数据处理效率。

参考文献

[1] 王耀南, 安果维, 王传成,等. 智能无人系统技术应用与发展趋势[J]. 中国舰船研究, 2022,17(5):9-26.

[2] 张云峰. 基于策略的分布式实时计算系统框架的设计和实现[D]. 济南:山东大学, 2019.

[3] 张智, 江果, 蒋鸣远. 面向军用网格的广域分布式数据处理框架[J]. 中国电子科学研究院学报, 2019,14(1):20-25.

[4] RÉVAY M, LIŠKA M. OODA loop in command & control systems [C] // 2017 Communication and Information Technologies (KIT). Vysoke Tatry, Slovak Republic: IEEE, 2017:1-4.

[5] 戴华东, 易晓东, 王彦臻, 等. 可持续自主学习的 micROS 机器人操作系统平行学习架构[J]. 计算机研究与发展, 2019,56(1):49-57.

[6] 邹立岩, 张明智, 柏俊汝. 一种基于 OODA—L 的智能无人集群作战仿真建模框架[C]// 2019 中国系统仿真与虚拟现实技术高层论坛论文集, 2019:241-248.

[7] 杨伟. 关于未来战斗机发展的若干讨论[J]. 航空学报, 2020,41(6):1-12.

[8] 王耀南. 人工智能赋能无人系统[J]. 智能系统学报, 2021,16(1):6.

[9] 韩银和,王展,郝沁汾. 构建基于 OODA 的专用智能计算机系统[EB/OL]. (2021-01-01) [2023-04-18]. https://dl.ccf.org.cn/article/articleDetail.html?id=5269586609965056.

[10] 张思齐, 沈钧戈, 郭行,等. 智能无人系统改变未来[J]. 无人系统技术, 2018(3):1-7.

[11] 李风雷, 卢昊, 宋闯, 等. 智能化战争与无人系统技术的发展[J]. 无人系统技术, 2018(2):14-23.

[12] BORTHAKUR D. The hadoop distributed file system: architecture and design [J]. Hadoop Project Website, 2007,11:1-10.

[13] ZAHARIA M, XIN R S, WENDELL P, et al. Apache spark: a unified engine for big data processing[J]. Communications of the ACM, 2016,59(11):56-65.

[14] ZAHARIA M, DAS T, LI H, et al. Discretized streams: fault-tolerant streaming computation at scale[C] // Proceedings of the 24th ACM Symposium on Operating Systems Principles. Farminton, USA: ACM, 2013: 423-438.

[15] EVANS R. Apache storm, a hands on tutorial[C]//2015 IEEE International Conference on Cloud Engineering. Tempe, USA: IEEE, 2015:2.

[16] CARBONE P, KATSIFODIMOS A, EWEN S, et al. Apache flink: stream and batch processing in a single engine[J]. The Bulletin of the Technical Committee on Data Engineering, 2015,38(4):28-38.

[17] 戚红雨. 流式处理框架发展综述[J]. 信息化研究, 2019,45(6):1-8.

[18] SPECIFICATION O M G A. Data distribution service for real-time systems version 1.2[M]. OMG: Object Management Group, 2007.

[19] 刘浩, 蒋泽军, 陈进朝. 一个面向服务的 DCPS 信息库的设计与实现[J]. 航空计算技术, 2013,43(5):100-103.

[20] 吕云飞, 王旋, 张军. 基于 DDS 技术的 JC 平台综合管理系统[J]. 船舶科学技术, 2011,33(5):47-52,62.

[21] 李海浩, 顾斌兵. DDS 在舰载 ZB 系统中的应用研究[J]. 计算机与数字工程, 2015,43(9):4-7.

[22] LV X J,LI X,YANG T, et al. QOS-aware publish-subscribe service for real-time data acquisition[C] // Business Intelligence for the Real-Time Enterprise: Second International Workshop, BIRTE 2008. Auckland, New Zealand: Springer Berlin Heidelberg, 2009:29-44.

[23] REDMON J, FARHADI A. YOLO v3: an incremental improvement[EB/OL]. (2018-04-08)[2023-04-12]. https://arxiv.org/pdf/1804.02767.

[24] REN S, HE K, GIRSHICK R, et al. Faster R-CNN: towards real-time object detection with region proposal networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017,39(6):1137-1149.

[25] HE K, ZHANG X, REN S, et al. Deep residual learning

for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA: IEEE, 2016:770-778.

[26] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition [EB/OL]. (2014-09-04) [2023-04-18]. http://arvix.org/pdf/1409.1556.

[27] JIA Y, SHELHAMER E, DONAHUE J, et al. Caffe: convolutional architecture for fast feature embedding[C]//Proceedings of the 22nd ACM International Conference on Multimedia. Orlando, USA: ACM, 2014:675-678.

[28] ABADI M, BARHAM P, CHEN J, et al. Tensorflow: a system for large-scale machine learning[C]// Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation. Savannah, USA: USENIX Association, 2016:265-283.

[29] CHEN T, LI M, LI Y, et al. MxNet: a flexible and efficient machine learning library for heterogeneous distributed systems [EB/OL]. (2015-10-03) [2023-04-18]. http://arvix.org/pdf/1512.01274.

OODAFlow: streaming data processing framework for intelligent unmanned systems

QUAN Zhenyu[*,**], YIN Longxiang[*], CHEN Xiaoming[*], HAN Yinhe[*]

([*] Research Center for Intelligent Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([**] University of Chinese Academy of Sciences, Beijing 100190)

Abstract

Intelligent unmanned system is a type of computing system that can autonomously perform real-time inferences and decisions, and formulate action plans in complex environments. The key to realizing real-time decision-making in intelligent unmanned systems lies in the real-time processing of streaming data. However, with the rapid development of artificial intelligence technology and sensor technology, the scale of data that intelligent unmanned systems need to process is constantly increasing, and the types of data are becoming more complex. Faced with the growing demand for data processing performance, intelligent unmanned systems need a well-optimized dedicated streaming data processing framework to enhance their data processing performance. To address this problem, this article proposes a stream data processing framework for intelligent unmanned systems called OODAFlow, which integrates the hardware features of intelligent unmanned systems and the data features of intelligent computing tasks with the observe-orient-decide-act (OODA) model to enable functions such as OODA task creation, task scheduling, and resource scheduling, it can schedule heterogeneous resources of intelligent unmanned systems and process intelligent computing tasks. This paper builds an OODA task processing system on intelligent unmanned systems to verify the feasibility of OODAFlow framework. Then, by proposing methods such as image preprocessing optimization, pipeline optimization, and orient node multi-process parallel optimization, the data throughput performance and resource utilization of the system have been improved. The experimental results in unmanned aerial vehicle intelligent control tasks show that when the proposed OODAFlow framework is used, the data processing performance of the intelligent unmanned system is improved by 73 times.

Key words: intelligent unmanned system, deep learning accelerator card, observe-orient-decide-act (OODA), streaming data processing framework, heterogeneous computing resource