



(21) 申请号 202211095125.8

G06F 8/76 (2018.01)

(22) 申请日 2022.09.05

G06N 3/08 (2023.01)

G06F 8/41 (2018.01)

(65) 同一申请的已公布的文献号

申请公布号 CN 115409166 A

(56) 对比文件

CN 102207887 A, 2011.10.05

CN 111667061 A, 2020.09.15

CN 111722875 A, 2020.09.29

(43) 申请公布日 2022.11.29

(73) 专利权人 青岛海洋科技中心

地址 266237 山东省青岛市即墨区问海中路168号

审查员 卢素斋

(72) 发明人 王勇 魏志强 马广浩 王玉成
周生辉 刘园园

(74) 专利代理机构 青岛联信知识产权代理事务
所(普通合伙) 37227

专利代理师 潘晋祥

(51) Int. Cl.

G06N 3/063 (2023.01)

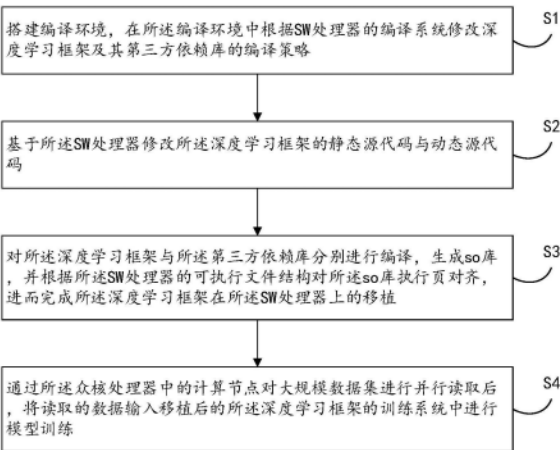
权利要求书2页 说明书11页 附图5页

(54) 发明名称

基于国产SW处理器的飞桨深度学习框架移植方法

(57) 摘要

本发明公开了一种基于国产SW处理器的飞桨深度学习框架移植方法,上述方法包括:搭建编译环境,在编译环境中根据SW处理器的编译系统修改深度学习框架及其第三方依赖库的编译策略;基于SW处理器修改深度学习框架的静态源代码与动态源代码;对深度学习框架与第三方依赖库分别进行编译,生成so库,并根据SW处理器的可执行文件结构对so库执行页对齐,进而完成深度学习框架在SW处理器上的移植。本方法可以将飞桨深度学习框架移植到基于国产SW处理器的神威超算机器,可满足更多的应用场景需求,并且代码开源,有完全自主知识产权,可持续优化和改进。



1. 一种基于国产SW处理器的飞桨深度学习框架移植方法,其特征在于,包括:

编译策略修改步骤:搭建编译环境,在所述编译环境中根据SW处理器的编译系统修改深度学习框架及其第三方依赖库的编译策略;

源代码修改步骤:基于所述SW处理器修改所述深度学习框架的静态源代码与动态源代码;

编译移植步骤:对所述深度学习框架与所述第三方依赖库分别进行编译,生成so库,并根据所述SW处理器的可执行文件结构对所述so库执行页对齐,进而完成所述深度学习框架在所述SW处理器上的移植;

还包括:

框架优化步骤:通过众核处理器中的计算节点对大规模数据集进行并行读取后,将读取的数据输入移植后的所述深度学习框架的训练系统中进行模型训练;

所述框架优化步骤包括:

数据量分析步骤:通过所述众核处理器中的控制节点分析获得所述大规模数据集的数据量以及当前空闲的计算节点的数量;

数据大小获得步骤:根据所述大规模数据集的数据量、当前空闲的计算节点的数量以及每一计算节点包含的主核数确定每一主核需读取的数据的大小;

模型训练步骤:各计算节点中的主核根据所述主核需读取的数据的大小从所述大规模数据集中读取数据,并将读取的数据输入所述训练系统进行模型训练;

所述模型训练步骤还包括:每一计算节点的各个主核将数据读入内存交叉段空间,训练完毕后,将所述内存交叉段空间内的数据进行合并,并随机排序,将随机排序后的数据按照所述主核需读取的数据的大小分配至每一主核进行下一次训练。

2. 根据权利要求1所述的飞桨深度学习框架移植方法,其特征在于,所述编译策略修改步骤包括:

编译环境搭建步骤:通过源码编译的方式安装环境软件;

编译控制FEATURE定义步骤:在所述深度学习框架的编译文件中定义编译控制FEATURE;

环境软件调用路径修改步骤:根据本地环境搭建并修改所述环境软件在编译时的调用路径;

编译参数修改步骤:在所述深度学习框架的编译文件中,根据所述SW处理器的编译选项与编译参数修改深度学习框架中的编译选项与编译参数;

编译控制FEATURE关闭步骤:关闭所述编译控制FEATURE中与SW处理器无关的硬件编译控制FEATURE和无法匹配的编译控制FEATURE。

3. 根据权利要求2所述的飞桨深度学习框架移植方法,其特征在于,所述编译策略修改步骤还包括:

第三方库路径定义步骤:在所述深度学习框架的编译文件中定义第三方依赖库的下载路径与第三方依赖库的编译路径;

第三方库编译文件修改步骤:在所述第三方依赖库的编译路径中新建第三方依赖库的编译文件,并基于所述SW处理器的编译系统修改所述第三方依赖库的编译文件。

4. 根据权利要求3所述的飞桨深度学习框架移植方法,其特征在于,所述第三方库编译

文件修改步骤还包括：在所述第三方依赖库的编译文件中修改第三方依赖库的编译文件路径、第三方依赖库路径以及第三方依赖库的编译指令和编译参数。

5. 根据权利要求1所述的飞桨深度学习框架移植方法，其特征在于，所述源代码修改步骤包括：将与硬件强相关的动态源代码通过bsub指令在所述SW处理器的计算节点上进行编译，动态生成与SW处理器强相关的动态源代码。

6. 根据权利要求5所述的飞桨深度学习框架移植方法，其特征在于，所述源代码修改步骤还包括：通过定义源码控制FEATURE控制基于SW处理器的源代码修改。

7. 根据权利要求1所述的飞桨深度学习框架移植方法，其特征在于，所述框架优化步骤还包括：当所述控制节点检测到主核的训练次数到达预设次数后，重新执行所述模型训练步骤。

基于国产SW处理器的飞桨深度学习框架移植方法

技术领域

[0001] 本发明涉及高性能计算技术领域,尤其涉及一种基于国产SW处理器的飞桨深度学习框架移植方法。

背景技术

[0002] 人工智能(Artificial Intelligence)是新一轮科技革命和产业变革的重要驱动力,深度学习(Deep Learning)已成为其中最具有影响的关键共性技术。在过去的十几年中,深度学习不论是在技术进步还是应用落地方面,都取得突破性进展。深度学习方法带动了语音、计算机视觉、自然语言处理等人工智能相关技术领域的跨越式发展,也在诸多行业应用中取得巨大成功。当下,深度学习相关的技术研究仍然持续火热,深度学习也正推动人工智能进入工业大生产时代。而作为深度学习技术和应用的底层基础设施,深度学习框架和平台的重要性也日益凸现,引发各界广泛关注。

[0003] 深度学习和应用的快速发展,迫切需要通过编程框架和开发平台来提升效率。深度学习框架下接芯片,可以屏蔽底层硬件差异并提供良好的执行性能;上承应用,成为算法研发和落地的标准环境。近年来,国外的大型科技公司纷纷在该领域布局,谷歌的TensorFlow和脸书的PyTorch已在开源社区取得广泛影响力。国内最早开源并且影响力最大的是百度推出的飞桨(PaddlePaddle),现已发展成为全面开源开放、技术领先、功能完备的产业级深度学习平台。飞桨框架基于和编程更加一致的深度学习计算抽象设计以及对应的前后端设计,做到了高效和易用的结合,具备易学易用的前端编程界面、统一高效的内部核心架构、原生完备的分布式训练支持和高性能可扩展的推理引擎。飞桨框架自然完备兼容命令式和声明式两种编程范式,并在大规模分布式训练、多端部署和产业级开源模型库等方面具备显著优势,是国内领先的成熟稳定、具备大规模推广条件的深度学习平台。

[0004] 超级计算机是衡量一个国家科技发展水平和综合国力的重要标志,其具有功能强,运算速度快,存储容量大的特点,支持大型复杂应用课题的计算,多用于国家高科技领域和尖端技术研究。“神威·太湖之光”是世界上首次性能超过100PFlops的超级计算机,也是我国首台具有完全自主知识产权的超级计算机,目前计算能力在TOP500世界排名第三(2016-2018连续三年名列榜首)、国内排名第一的超级计算机,峰值性能9.3亿亿次,搭载了国产SW26010异构众核处理器,该处理器使用64位自主申威指令集,采用主-从核架构,如图1所示,一个处理器包含4个核组,每个核组包含1个主核、1个8*8从核阵列和1个内存控制器,单处理器峰值计算能力为3TFlops,访存带宽为130GB/s。“神威·太湖之光”超级计算机总共包含40960个SW26010异构众核处理器。

[0005] 深度学习是驱动人工智能近年来取得突破的关键核心技术,带动了语音、视觉、自然语言相关技术领域的跨越式发展,加速推动了人工智能的产业化落地。深度学习框架和平台作为深度学习底层基础设施,对深度学习技术和应用的快速发展起到了重要推动作用。神威超算是我国完全具有自主知识产权的超级计算机,国产神威超算现有的机器学习框架包括Facebook的Pytorch和Google的TensorFlow,虽然这两大机器学习框架使用人数

也比较多,但是在神威超算上存在明显不足:

[0006] 1、神威超算现有的两大机器学习框架版本落后,Pytorch的神威超算版本是1.5.0,而Pytorch最新的官方版本是1.10.1,TensorFlow的神威超算版本是2.3.0,而其官方最新版本是2.7.0,版本普遍落后,无法满足新的应用需求;

[0007] 2、神威超算现有的两大机器学习框架未开源,关键源代码已经封装成库文件,无法进行持续优化演进;

[0008] 3、神威超算现有的两大机器学习框架均由国外厂商开发,无自主知识产权;

[0009] 4、在大规模数据进行训练时,没有针对国产SW处理器进行内存和数据抽样的优化。

发明内容

[0010] 本发明针对上述神威超算现有的机器学习框架版本落后且无法优化的技术缺陷,提出一种基于国产SW处理器的飞桨深度学习框架移植方法,包括:

[0011] 编译策略修改步骤:搭建编译环境,在所述编译环境中根据SW处理器的编译系统修改深度学习框架及其第三方依赖库的编译策略;

[0012] 源代码修改步骤:基于所述SW处理器修改所述深度学习框架的静态源代码与动态源代码;

[0013] 编译移植步骤:对所述深度学习框架与所述第三方依赖库分别进行编译,生成so库,并根据所述SW处理器的可执行文件结构对所述so库执行页对齐,进而完成所述深度学习框架在所述SW处理器上的移植。

[0014] 上述飞桨深度学习框架移植方法,其中,还包括:

[0015] 框架优化步骤:通过所述众核处理器中的计算节点对大规模数据集进行并行读取后,将读取的数据输入移植后的所述深度学习框架的训练系统中进行模型训练。

[0016] 上述飞桨深度学习框架移植方法,其中,所述编译策略修改步骤包括:

[0017] 编译环境搭建步骤:通过源码编译的方式安装环境软件;

[0018] 编译控制FEATURE定义步骤:在所述深度学习框架的编译文件中定义编译控制FEATURE;

[0019] 环境软件调用路径修改步骤:根据本地环境搭建并修改所述环境软件在编译时的调用路径;

[0020] 编译参数修改步骤:在所述深度学习框架的编译文件中,根据所述SW处理器的编译选项与编译参数修改深度学习框架中的编译选项与编译参数;

[0021] 编译控制FEATURE关闭步骤:关闭所述编译控制FEATURE中与SW处理器无关的硬件编译控制FEATURE和无法匹配的编译控制FEATURE。

[0022] 上述飞桨深度学习框架移植方法,其中,所述编译策略修改步骤还包括:

[0023] 第三方库路径定义步骤:在所述深度学习框架的编译文件中定义第三方依赖库的下载路径与第三方依赖库的编译路径;

[0024] 第三方库编译文件修改步骤:在所述第三方依赖库的编译路径中新建第三方依赖库的编译文件,并基于所述SW处理器的编译系统修改所述第三方依赖库的编译文件。

[0025] 上述飞桨深度学习框架移植方法,其中,所述第三方库编译文件修改步骤还包括:

在所述第三方依赖库的编译文件中修改第三方依赖库的编译文件路径、第三方依赖库路径以及第三方依赖库的编译指令和编译参数。

[0026] 上述飞桨深度学习框架移植方法,其中,所述源代码修改步骤包括:将与硬件强相关的动态源代码通过bsub指令在所述SW处理器的计算节点上进行编译,动态生成与SW处理器强相关的动态源代码。

[0027] 上述飞桨深度学习框架移植方法,其中,所述源代码修改步骤还包括:通过定义源码控制FEATURE控制基于SW处理器的源代码修改。

[0028] 上述飞桨深度学习框架移植方法,其中,所述框架优化步骤包括:

[0029] 数据量分析步骤:通过所述众核处理器中的控制节点分析获得所述大规模数据集的数据量以及当前空闲的计算节点的数量;

[0030] 数据大小获得步骤:根据所述大规模数据集的数据量、当前空闲的计算节点的数量以及每一计算节点包含的主核数确定每一主核需读取的数据的大小;

[0031] 模型训练步骤:各计算节点中的主核根据所述主核需读取的数据的大小从所述大规模数据集中读取数据,并将读取的数据输入所述训练系统进行模型训练。

[0032] 上述飞桨深度学习框架移植方法,其中,所述模型训练步骤还包括:每一计算节点的各个主核将数据读入内存交叉段空间,训练完毕后,将所述内存交叉段空间内的数据进行合并,并随机排序,将随机排序后的数据按照所述主核需读取的数据的大小分配至每一主核进行下一次训练。

[0033] 上述飞桨深度学习框架移植方法,其中,所述框架优化步骤还包括:当所述控制节点检测到主核的训练次数到达预设次数后,重新执行所述模型训练步骤。

[0034] 与现有技术相比,本发明的优点和积极效果在于:

[0035] 1、本发明将百度飞桨深度学习框架成功移植到了基于国产SW处理器的神威超计算机上,可满足更多的应用场景需求,并且代码开源,有完全自主知识产权,可持续优化和改进,以及,飞桨深度学习框架由国内厂商百度研发,核心技术不用受制于国外厂商,有完全自主知识产权;

[0036] 2、本发明提出的基于国产SW众核处理器的飞桨深度学习框架,可开展大规模并行模型训练和模型预测,加快机器学习模型训练和预测速度,有效缩短模型训练和预测时间,并在大规模数据进行并行训练时,兼顾内存优化的同时,考虑了数据集的随机抽样,使训练出的模型更加准确。

附图说明

[0037] 图1为本发明提供的SW26010异构众核处理器架构图;

[0038] 图2为本发明提供的基于国产SW处理器的飞桨深度学习框架移植方法的步骤示意图;

[0039] 图3为本发明提供的基于图2中步骤S1的流程示意图;

[0040] 图4为本发明提供的基于国产SW处理器的飞桨深度学习框架移植方法一实施例流程图示意图;

[0041] 图5为本发明提供的基于国产SW处理器的大规模数据集随机抽样框架图;

[0042] 图6为本发明提供的基于国产SW处理器的飞桨深度学习框架模型训练图;

[0043] 图7为本发明提供的一种基于国产SW处理器的飞桨深度学习框架移植系统的框架图;

[0044] 图8为根据本申请实施例的计算机设备的框架图;

[0045] 其中,附图标记为:

[0046] 11、编译策略修改单元;111、编译环境搭建模块;112、编译控制FEATURE定义模块;113、环境软件调用路径修改模块;114、编译参数修改模块;115、编译控制FEATURE关闭模块;116、第三方库路径定义模块;117、第三方库编译文件修改模块;12、源代码修改单元;13、编译移植单元;14、框架优化单元;81、处理器;82、存储器;83、通信接口;80、总线。

具体实施方式

[0047] 为了使本申请的目的、技术方案及优点更加清楚明白,以下结合附图及实施例,对本申请进行描述和说明。应当理解,此处所描述的具体实施例仅仅用以解释本申请,并不用于限定本申请。基于本申请提供的实施例,本领域普通技术人员在没有作出创造性劳动的前提下所获得的所有其他实施例,都属于本申请保护的范围。

[0048] 显而易见地,下面描述中的附图仅仅是本申请的一些示例或实施例,对于本领域的普通技术人员而言,在不付出创造性劳动的前提下,还可以根据这些附图将本申请应用于其他类似情景。此外,还可以理解的是,虽然这种开发过程中所作出的努力可能是复杂并且冗长的,然而对于与本申请公开的内容相关的本领域的普通技术人员而言,在本申请揭露的技术内容的基础上进行的一些设计,制造或者生产等变更只是常规的技术手段,不应理解为对本申请公开的内容不充分。

[0049] 在本申请中提及“实施例”意味着,结合实施例描述的特定特征、结构或特性可以包含在本申请的至少一个实施例中。在说明书中的各个位置出现该短语并不一定均是指相同的实施例,也不是与其它实施例互斥的独立的或备选的实施例。本领域普通技术人员显式地和隐式地理解的是,本申请所描述的实施例在不冲突的情况下,可以与其它实施例相结合。

[0050] 除非另作定义,本申请所涉及的技术术语或者科学术语应当为本申请所属技术领域内具有一般技能的人士所理解的通常意义。本申请所涉及的“一”、“一个”、“一种”、“该”等类似词语并不表示数量限制,可表示单数或复数。本申请所涉及的术语“包括”、“包含”、“具有”以及它们任何变形,意图在于覆盖不排他的包含;例如包含了一系列步骤或模块(单元)的过程、方法、系统、产品或设备没有限定于已列出的步骤或单元,而是可以还包括没有列出的步骤或单元,或可以还包括对于这些过程、方法、产品或设备固有的其它步骤或单元。本申请所涉及的“连接”、“相连”、“耦接”等类似的词语并非限于物理的或者机械的连接,而是可以包括电气的连接,不管是直接的还是间接的。本申请所涉及的“多个”是指两个或两个以上。“和/或”描述关联对象的关联关系,表示可以存在三种关系,例如,“A和/或B”可以表示:单独存在A,同时存在A和B,单独存在B这三种情况。字符“/”一般表示前后关联对象是一种“或”的关系。本申请所涉及的术语“第一”、“第二”、“第三”等仅仅是区别类似的对象,不代表针对对象的特定排序。

[0051] 下面结合附图所示的各实施方式对本发明进行详细说明,但应当说明的是,这些实施方式并非对本发明的限制,本领域普通技术人员根据这些实施方式所作的功能、方法、

或者结构上的等效变换或替代,均属于本发明的保护范围之内。

[0052] 在详细阐述本发明各个实施例之前,对本发明的核心发明思想予以概述,并通过下述若干实施例予以详细阐述。

[0053] 首先,根据国产神威超算的SW处理器编译系统,修改百度飞桨深度学习框架的编译策略,并搭建编译环境,将百度飞桨核心框架及其依赖的第三方关联库分别进行编译移植;其次,修改飞桨深度学习框架源代码,解决移植过程中出现的静态代码不匹配问题;再次,对于与硬件强相关的动态源代码,设计bsub指令和节点编译策略,动态生成与国产神威超算SW处理器强相关的动态源代码;完成百度飞桨深度学习框架在国产神威超算上的移植和构建;最后,实现飞桨深度学习框架在大规模数据集并行训练时的数据抽样优化。

[0054] 实施例一:

[0055] 图2为本发明提供的一种基于国产SW处理器的飞桨深度学习框架移植方法的步骤示意图。如图2所示,本实施例揭示了一种基于国产SW处理器的飞桨深度学习框架移植方法(以下简称“方法”)的具体实施方式。

[0056] 具体而言,本实施例所揭示的方法主要包括以下步骤:

[0057] 步骤S1:搭建编译环境,在所述编译环境中根据SW处理器的编译系统修改深度学习框架及其第三方依赖库的编译策略;

[0058] 在本实施例中,SW处理器为国产神威超算计算机搭载的SW处理器,所述深度学习框架为百度飞桨深度学习框架。

[0059] 具体而言,如图3所示,步骤S1具体包括以下包括:

[0060] 步骤S11:通过源码编译的方式安装环境软件;其中,环境软件包括python3.6、protoc3.19.00、CMake3.19和patchelf;

[0061] 步骤S12:在所述深度学习框架的编译文件中定义编译控制FEATURE;其中,编译控制FEATURE用来总体控制适配SW处理器的编译修改和原有编译流程相互兼容;

[0062] 步骤S13:根据本地环境搭建并修改所述环境软件在编译时的调用路径;

[0063] 步骤S14:在所述深度学习框架的编译文件中,根据所述SW处理器的编译选项与编译参数修改深度学习框架中的编译选项与编译参数;

[0064] 由于国产神威超算编译器与原有的飞桨编译器的名称及参数不同,在所述深度学习框架的编译文件中将C++编译器的名称修改为swg++,将C编译器的名称修改为swgcc,将编译参数设置为SW编译器的相应参数。

[0065] 步骤S15:关闭所述编译控制FEATURE中与SW处理器无关的硬件编译控制FEATURE和无法匹配的编译控制FEATURE;避免在国产SW编译器上由于不匹配出现编译错误;

[0066] 步骤S16:在所述深度学习框架的编译文件中定义第三方依赖库的下载路径与第三方依赖库的编译路径;

[0067] 具体的,由于神威超算机器的网络限制,无法在线下载第三方依赖库,为避免出现网络原因导致的编译失败,在深度学习框架的编译文件中定义第三方依赖库的下载路径,此路径为本地路径,在此路径中保存已经下载好的第三方依赖库供飞桨框架编译使用,其中,第三方依赖库包括boost,pocketfft,xxhash,gloo等;

[0068] 步骤S17:在所述第三方依赖库的编译路径中新建第三方依赖库的编译文件,并基于所述SW处理器的编译系统修改所述第三方依赖库的编译文件。

[0069] 步骤S17还包括:在所述第三方依赖库的编译文件中修改第三方依赖库的编译文件路径、第三方依赖库路径以及第三方依赖库的编译指令和编译参数。

[0070] 具体来说,在第三方依赖库的编译文件中,修改第三方依赖库的编译文件路径,其中各个第三方依赖库的编译文件存放于此路径下,这样做有利于飞桨原始框架的保护,使得基于国产SW处理器的编译流程更加清晰;

[0071] 在第三方依赖库的编译文件中,将用于存放从下载路径获取的第三方依赖库的第三方依赖库路径,修改为SW处理器单独的路径;

[0072] 在第三方依赖库的编译文件中,将第三方依赖库的编译指令和编译参数修改为国产SW处理器对应的指令和参数。

[0073] 步骤S2:基于所述SW处理器修改所述深度学习框架的静态源代码与动态源代码;

[0074] 具体来说,基于国产SW处理器,修改飞桨深度学习框架的静态源代码,能够解决移植过程中出现的静态代码不匹配问题,比如由于编译器不同,导致的函数接口不一致问题,结构体变量定义不一致问题等。另外,考虑到源码多平台的兼容性,定义源码FEATURE,总体控制基于SW处理器的源代码修改;

[0075] 其中,步骤S2还包括:根据队列名称、计算节点数、共享模式以及从核数设计bsub指令,将与硬件强相关的动态源代码通过bsub指令在SW处理器的计算节点上进行编译,动态生成与SW处理器强相关的动态源代码。

[0076] 步骤S3:对所述深度学习框架与所述第三方依赖库分别进行编译,生成so库,并根据所述SW处理器的可执行文件结构对所述so库执行页对齐,进而完成所述深度学习框架在所述SW处理器上的移植。

[0077] 基于国产SW众核处理器的最新版本飞桨深度学习框架,可满足更多的应用场景需求,并且代码开源,有完全自主知识产权,可持续优化和改进。在国产神威超算机器上,可基于此框架开展大规模并行模型训练和模型预测,加快机器学习模型训练和预测速度,有效缩短模型训练和预测时间。

[0078] 进一步的,在一些实施例中,还包括:

[0079] 步骤S4:通过所述众核处理器中的计算节点对大规模数据集进行并行读取后,将读取的数据输入移植后的所述深度学习框架的训练系统中进行模型训练。

[0080] 具体地,步骤S4具体包括以下内容:

[0081] 步骤S41:通过所述众核处理器中的控制节点分析获得所述大规模数据集的数据量以及当前空闲的计算节点的数量;

[0082] 步骤S42:根据所述大规模数据集的数据量、当前空闲的计算节点的数量以及每一计算节点包含的主核数确定每一主核需读取的数据的大小;

[0083] 步骤S43:各计算节点中的主核根据所述主核需读取的数据的大小从所述大规模数据集中读取数据,并将读取的数据输入所述训练系统进行模型训练;如此,计算节点的主核从大规模数据集中并行读取一定大小的数据,避免了单个主核读取数据量过大而出现内存不足的情况。

[0084] 以及,一次训练完成后,需要进行参数同步,控制节点收集各个计算节点的梯度信息,将收集的梯度信息进行加权平均计算后,将结果广播给各个计算节点,各个计算节点主核根据该结果进行模型参数的更新,计算节点主核更新模型参数后,开始下一次的训练。

[0085] 其中,步骤S43还包括:每一计算节点的各个主核将数据读入内存交叉段空间,训练完毕后,将所述内存交叉段空间内的数据进行合并,并随机排序,将随机排序后的数据按照所述主核需读取的数据的大小分配至每一主核进行下一次训练。其中,每个计算节点中的主核的交叉段内存空间是共享的,考虑到数据集抽样的随机性,所以需要将内存交叉段内的数据进行合并与随机排序后进行下一次训练,这样可以在最大程度减少IO开销的同时,兼顾节点内数据抽样的随机性;

[0086] 最后,步骤S4还包括:当所述控制节点检测到主核的训练次数到达预设次数后,重新执行所述模型训练步骤。直至训练结束。

[0087] 以下,请参照图4。图4为本发明提供的一种基于国产SW处理器的飞桨深度学习框架移植方法一实施例流程示意图;结合图4,具体说明本方法的应用流程如下:

[0088] Step1:搭建编译环境,使用源码编译的方式安装python3.6、protoc3.19.00、CMake3.19和patchelf;

[0089] Step2:由于国产神威超算和X86的硬件架构的差异性,需要基于国产神威超算的SW处理器编译系统,修改百度飞桨深度学习框架的编译策略,使得飞桨深度学习框架能够在国产神威超算上成功进行编译。考虑到编译策略的跨平台通用性,在飞桨框架现有的编译文件CMakeLists.txt中定义控制FEATURE (BUILD_SW_MAKE),用来总体控制适配SW处理器的编译修改和原有编译流程相互兼容;

[0090] Step3:根据本地环境搭建路径,修改编译时环境软件的调用路径,比如python软件;

[0091] Step4:修改编译选项和参数:由于国产神威超算编译器与原有的飞桨编译器的名称及参数不同,在CMakeLists.txt文件中,修改C++编译器名称为swg++,修改C编译器名称为swgcc,编译参数设置为SW编译器相应参数,比如:-mhost:指示SW处理器生成主核代码;-mftz:指示编译器在发生下溢时将真零写入目标寄存器;-mieee:指示编译器允许使用产生精确结果的任何操作;-msimd:指示编译器支持SIMD扩展编程接口,生成相应的SIMD扩展指令;-fPIC:编译为库文件等,使得飞桨框架的编译参数与国产神威处理器相匹配;

[0092] Step5:关闭与SW处理器无关的硬件编译控制FEATURE和无法匹配的编译控制FEATURE,避免在国产神威编译器上不匹配出现编译错误(注:FEATURE表示控制编译某项功能的开关);

[0093] Step6:由于神威超算机器的网络限制,无法在线下载第三方依赖库,为避免出现网络原因导致的编译失败,在CMakeLists.txt文件中定义第三方依赖库的下载路径为LOCAL_3RD_ROOT,此路径为本地路径,在此路径中保存已经下载好的第三方依赖库供飞桨框架编译使用,第三方依赖库包括boost,pocketfft,xxhash,gloo等;

[0094] Step7:第三方依赖库的编译修改。为适配国产神威处理器的编译,第三方依赖库需要进行相应的编译修改,由于涉及依赖库较多,为保护原始框架流程,与原始飞桨框架进行区分,新建第三方依赖库编译路径third_party_sw,并在CMakeLists.txt文件中引入第三方依赖库的编译路径为third_party_sw;

[0095] Step8:在third_party_sw路径中,新建第三方库编译文件thrid_parth_sw.cmake,并在此文件中修改第三方依赖库的编译文件路径为external_sw,各个第三方依赖库的SW编译器编译文件存放于此路径下,这样做有利于飞桨原始框架的保护,使得基于

国产SW处理器的编译流程更加清晰；

[0096] Step9:基于国产SW处理器编译系统,修改各个第三方依赖库的编译文件,包括:修改第三方依赖库路径,第三方依赖库路径修改为国产SW处理器单独的路径;修改编译指令和参数,修改为国产SW处理器对应的指令和参数,比如-mhost编译生成主核代码等;

[0097] Step10:基于国产SW处理器,修改飞桨深度学习框架源代码,解决移植过程中出现的静态代码不匹配问题。比如由于编译器不同,导致的函数接口不一致问题,结构体变量定义不一致问题等。另外,考虑到源码多平台的兼容性,定义源码控制FEATURE (BUILD_SW_CODE),总体控制基于SW处理器的源代码修改;

[0098] Step11:对于与硬件强相关的动态源代码,根据队列名称、计算节点数、共享模式、从核数,设计bsub指令,将硬件强相关的编译通过bsub指令提交到国产SW处理器计算节点上进行编译,动态生成与国产SW处理器强相关的动态源代码,包括Python代码和C/C++代码;

[0099] Step12:编译完成,由于国产SW处理器统一执行8k页对齐,需要对编译生成的so库执行页对齐PAGE_ALIGN,由4k对齐到8k,匹配国产SW处理器可执行文件结构,才能成功运行;

[0100] Step13:成功实现百度飞桨框架在国产神威超算机器上的移植。最后,测试基于国产SW处理器的飞桨深度学习框架是否能够成功运行,进行如图6所示的模型训练。

[0101] 另一方面,本实施例还提供飞桨深度学习框架在大规模数据集并行训练时的数据抽样优化方法。以下,请参照图5。图5为本发明提供的基于国产SW处理器的大规模数据集随机抽样框架图;数据流向可参考图5,其主要步骤如下:

[0102] Step1:启动SW处理器的控制节点,控制节点分析大规模数据集的数据量和当前空闲计算节点的资源数目,确定一个large_batch的大小m,其中,large_batch的大小m与空闲计算节点的数量相关,可以通过数据量与空闲计算节点数量的比值获得;

[0103] Step2:SW处理器的每个计算节点(CPU)中包含4个主核,控制节点将一个large_batch的大小值m广播给各个计算节点,即把m/4值广播给每个主核,把m/4定义为batch;

[0104] Step3:各个计算节点中的主核收到batch的大小n,从大规模数据集中并行读取n大小的数据,避免了单个主核读取数据量过大而出现内存不足;

[0105] Step4:计算节点中的主核将各自的数据读入内存交叉段空间,每个计算节点中4个主核的交叉段内存空间是共享的;

[0106] Step5:每个主核将各自的数据输入飞桨深度学习框架的训练系统开始模型训练,训练系统中每次输入数据集的大小为mini_batch;

[0107] Step6:一次训练完成后,需要进行参数同步,控制节点负责收集各个计算节点的梯度信息;

[0108] Step7:控制节点将收集的梯度信息进行加权平均计算,然后将结果广播给各个计算节点,各个计算节点主核进行模型参数的更新;

[0109] Step8:计算节点主核更新模型参数后,开始下一次的训练;

[0110] Step9:一个计算节点内的4个主核训练完毕后,考虑到batch数据集抽样的随机性,需要将内存交叉段内的数据进行合并,并将数据随机排序,重新按照batch大小分配给一个计算节点内的4个主核进行训练,这样可以在最大程度减少IO开销的同时,兼顾节点内

数据抽样的随机性;

[0111] Step10:控制节点检测到主核训练到一定次数后,考虑到large_batch数据集的抽样随机性,可重新进行各个计算节点对大规模数据集的并行读取;

[0112] Step11:直至最后训练结束。

[0113] 本发明克服了现有技术中的缺点,提供了一种基于国产SW处理器的飞桨深度学习框架移植方法。该方法将百度飞桨(PaddlePaddle)深度学习框架成功移植到了基于国产SW处理器的神威超算机器上,移植的飞桨版本号为最新稳定版2.2,可满足最新的应用需求,并且所有代码开源,可持续优化改进,另外,飞桨由国内厂商百度研发,核心技术不用受制于国外厂商,有完全自主知识产权。基于国产SW众核处理器的飞桨深度学习框架,可开展大规模并行模型训练和模型预测,加快机器学习模型训练和预测速度,有效缩短模型训练和预测时间。在大规模数据进行并行训练时,兼顾内存优化的同时,考虑了数据集的随机抽样,使训练出的模型更加准确。

[0114] 实施例二:

[0115] 结合实施例一所揭示的一种基于国产SW处理器的飞桨深度学习框架移植方法,本实施例揭示了一种基于国产SW处理器的飞桨深度学习框架移植系统(以下简称“系统”)的具体实施示例。

[0116] 参照图7所示,所述系统包括:

[0117] 编译策略修改单元11:搭建编译环境,在所述编译环境中根据SW处理器的编译系统修改深度学习框架及其第三方依赖库的编译策略;

[0118] 源代码修改单元12:基于所述SW处理器修改所述深度学习框架的静态源代码与动态源代码;

[0119] 编译移植单元13:对所述深度学习框架与所述第三方依赖库分别进行编译,生成so库,并根据所述SW处理器的可执行文件结构对所述so库执行页对齐,进而完成所述深度学习框架在所述SW处理器上的移植;

[0120] 框架优化单元14:通过所述众核处理器中的计算节点对大规模数据集进行并行读取后,将读取的数据输入移植后的所述深度学习框架的训练系统中进行模型训练。

[0121] 其中,所述编译策略修改单元11包括:

[0122] 编译环境搭建模块111:通过源码编译的方式安装环境软件;

[0123] 编译控制FEATURE定义模块112:在所述深度学习框架的编译文件中定义编译控制FEATURE;

[0124] 环境软件调用路径修改模块113:根据本地环境搭建并修改所述环境软件在编译时的调用路径;

[0125] 编译参数修改模块114:在所述深度学习框架的编译文件中,根据所述SW处理器的编译选项与编译参数修改深度学习框架中的编译选项与编译参数;

[0126] 编译控制FEATURE关闭模块115:关闭所述编译控制FEATURE中与SW处理器无关的硬件编译控制FEATURE和无法匹配的编译控制FEATURE;

[0127] 第三方库路径定义模块116:在所述深度学习框架的编译文件中定义第三方依赖库的下载路径与第三方依赖库的编译路径;

[0128] 第三方库编译文件修改模块117:在所述第三方依赖库的编译路径中新建第三方

依赖库的编译文件,并基于所述SW处理器的编译系统修改所述第三方依赖库的编译文件。

[0129] 本实施例所揭示的一种飞桨深度学习框架移植系统与实施例一所揭示的一种飞桨深度学习框架移植方法中其余相同部分的技术方案,请参照实施例一所述,在此不再赘述。

[0130] 实施例三:

[0131] 结合图8所示,本实施例揭示了一种计算机设备的一种具体实施方式。计算机设备可以包括处理器81以及存储有计算机程序指令的存储器82。

[0132] 具体地,上述处理器81可以包括中央处理器(CPU),或者特定集成电路(Application Specific Integrated Circuit,简称为ASIC),或者可以被配置成实施本申请实施例的一个或多个集成电路。

[0133] 其中,存储器82可以包括用于数据或指令的大容量存储器。举例来说而非限制,存储器82可包括硬盘驱动器(Hard Disk Drive,简称为HDD)、软盘驱动器、固态驱动器(Solid State Drive,简称为SSD)、闪存、光盘、磁光盘、磁带或通用串行总线(Universal Serial Bus,简称为USB)驱动器或者两个或更多个以上这些的组合。在合适的情况下,存储器82可包括可移除或不可移除(或固定)的介质。在合适的情况下,存储器82可在数据处理装置的内部或外部。在特定实施例中,存储器82是非易失性(Non-Volatile)存储器。在特定实施例中,存储器82包括只读存储器(Read-Only Memory,简称为ROM)和随机存取存储器(Random Access Memory,简称为RAM)。在合适的情况下,该ROM可以是掩模编程的ROM、可编程ROM(Programmable Read-Only Memory,简称为PROM)、可擦除PROM(Erasable Programmable Read-Only Memory,简称为EPROM)、电可擦除PROM(Electrically Erasable Programmable Read-Only Memory,简称为EEPROM)、电可改写ROM(Electrically Alterable Read-Only Memory,简称为EAROM)或闪存(FLASH)或者两个或更多个以上这些的组合。在合适的情况下,该RAM可以是静态随机存取存储器(Static Random-Access Memory,简称为SRAM)或动态随机存取存储器(Dynamic Random Access Memory,简称为DRAM),其中,DRAM可以是快速页模式动态随机存取存储器(Fast Page Mode Dynamic Random Access Memory,简称为FPMDRAM)、扩展数据输出动态随机存取存储器(Extended Data Out Dynamic Random Access Memory,简称为EDODRAM)、同步动态随机存取内存(Synchronous Dynamic Random-Access Memory,简称SDRAM)等。

[0134] 存储器82可以用来存储或者缓存需要处理和/或通信使用的各种数据文件,以及处理器81所执行的可能的计算机程序指令。

[0135] 处理器81通过读取并执行存储器82中存储的计算机程序指令,以实现上述实施例中的任意一种飞桨深度学习框架移植方法。

[0136] 在其中一些实施例中,计算机设备还可包括通信接口83和总线80。其中,如图8所示,处理器81、存储器82、通信接口83通过总线80连接并完成相互间的通信。

[0137] 通信接口83用于实现本申请实施例中各模块、装置、单元和/或设备之间的通信。通信接口83还可以实现与其他部件例如:外接设备、图像/数据采集设备、数据库、外部存储以及图像/数据处理工作站等之间进行数据通信。

[0138] 总线80包括硬件、软件或两者,将计算机设备的部件彼此耦接在一起。总线80包括但不限于以下至少之一:数据总线(Data Bus)、地址总线(Address Bus)、控制总线

(Control Bus)、扩展总线(Expansion Bus)、局部总线(Local Bus)。举例来说而非限制,总线80可包括图形加速接口(Accelerated Graphics Port,简称为AGP)或其他图形总线、增强工业标准架构(Extended Industry Standard Architecture,简称为EISA)总线、前端总线(Front Side Bus,简称为FSB)、超传输(Hyper Transport,简称为HT)互连、工业标准架构(Industry Standard Architecture,简称为ISA)总线、无线带宽(InfiniBand)互连、低引脚数(Low Pin Count,简称为LPC)总线、存储器总线、微信道架构(Micro Channel Architecture,简称为MCA)总线、外围组件互连(Peripheral Component Interconnect,简称为PCI)总线、PCI-Express(PCI-X)总线、串行高级技术附件(Serial Advanced Technology Attachment,简称为SATA)总线、视频电子标准协会局部(Video Electronics Standards Association Local Bus,简称为VLB)总线或其他合适的总线或者两个或更多个以上这些的组合。在合适的情况下,总线80可包括一个或多个总线。尽管本申请实施例描述和示出了特定的总线,但本申请考虑任何合适的总线或互连。

[0139] 另外,结合上述实施例中的飞桨深度学习框架移植方法,本申请实施例可提供一种计算机可读存储介质来实现。该计算机可读存储介质上存储有计算机程序指令;该计算机程序指令被处理器执行时实现上述实施例中的任意一种飞桨深度学习框架移植方法。

[0140] 以上所述实施例的各技术特征可以进行任意的组合,为使描述简洁,未对上述实施例中的各个技术特征所有可能的组合都进行描述,然而,只要这些技术特征的组合不存在矛盾,都应当认为是本说明书记载的范围。

[0141] 以上所述实施例仅表达了本申请的几种实施方式,其描述较为具体和详细,但并不能因此而理解为对发明专利范围的限制。应当指出的是,对于本领域的普通技术人员来说,在不脱离本申请构思的前提下,还可以做出若干变形和改进,这些都属于本申请的保护范围。因此,本申请专利的保护范围应以所附权利要求为准。

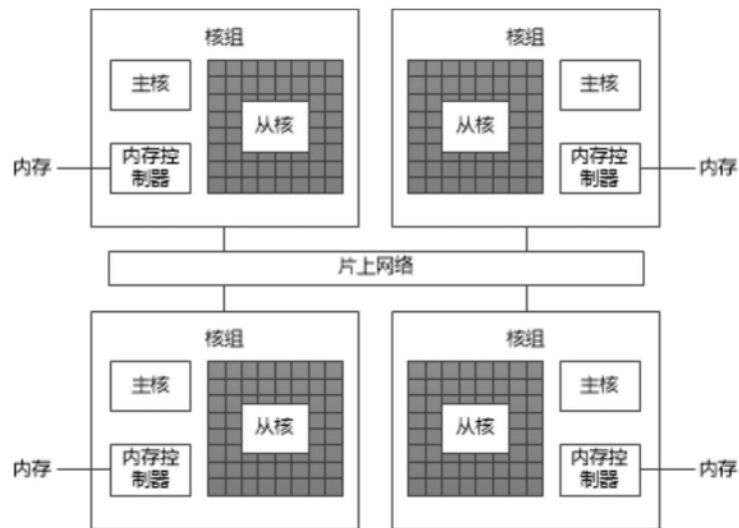


图1

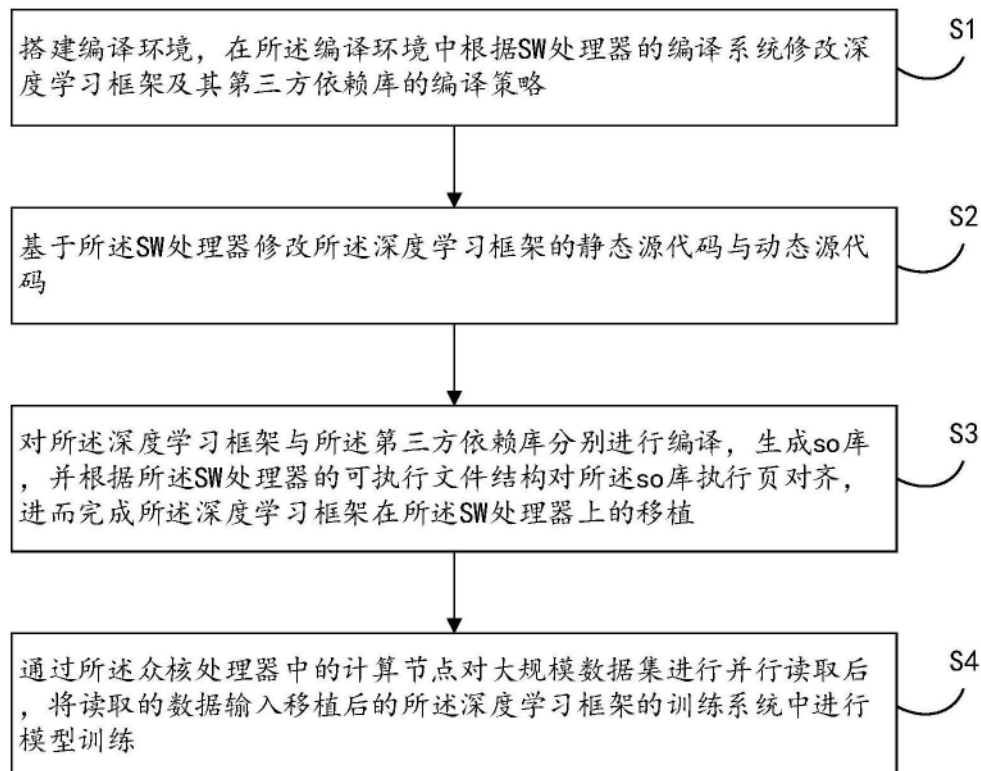


图2

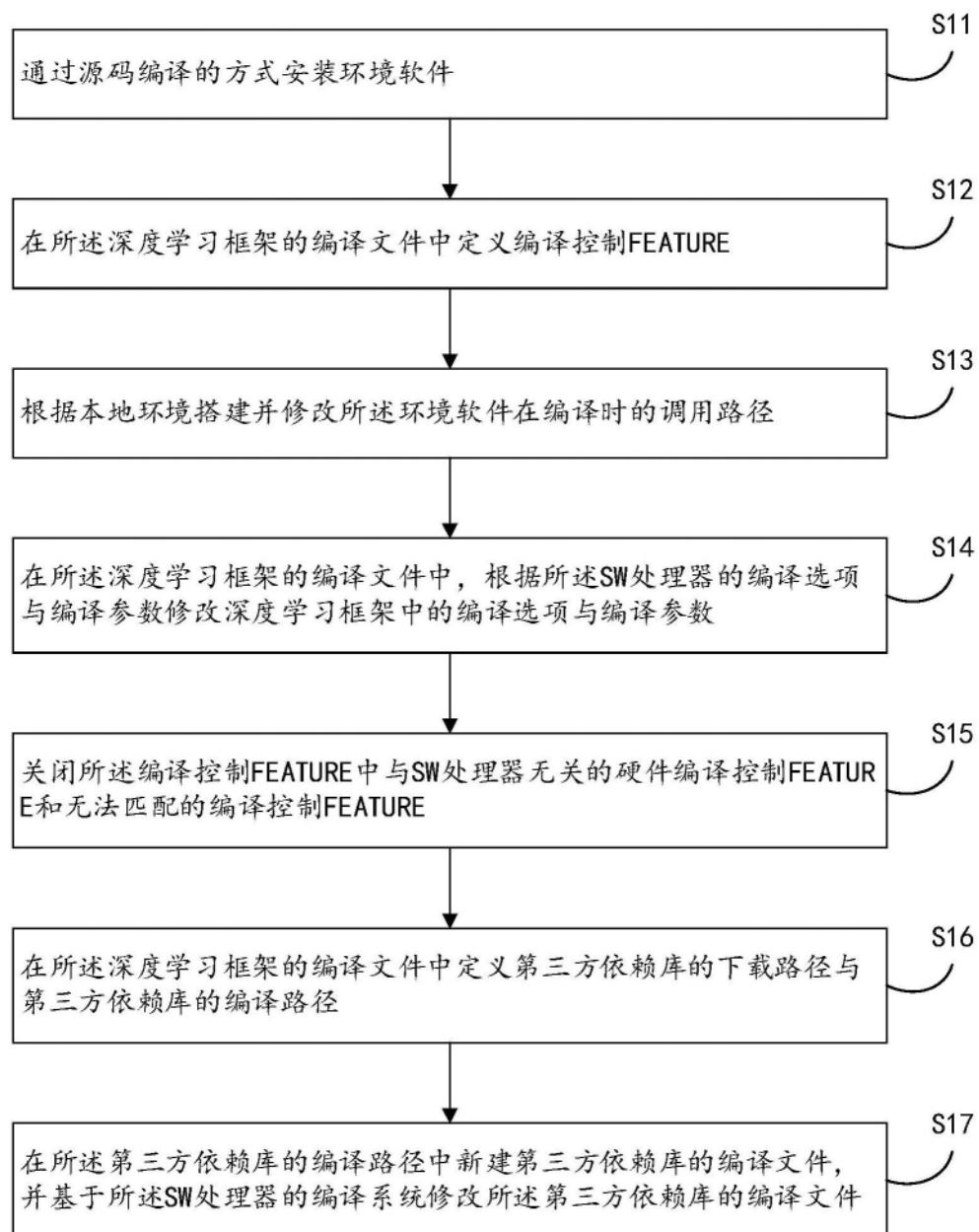


图3

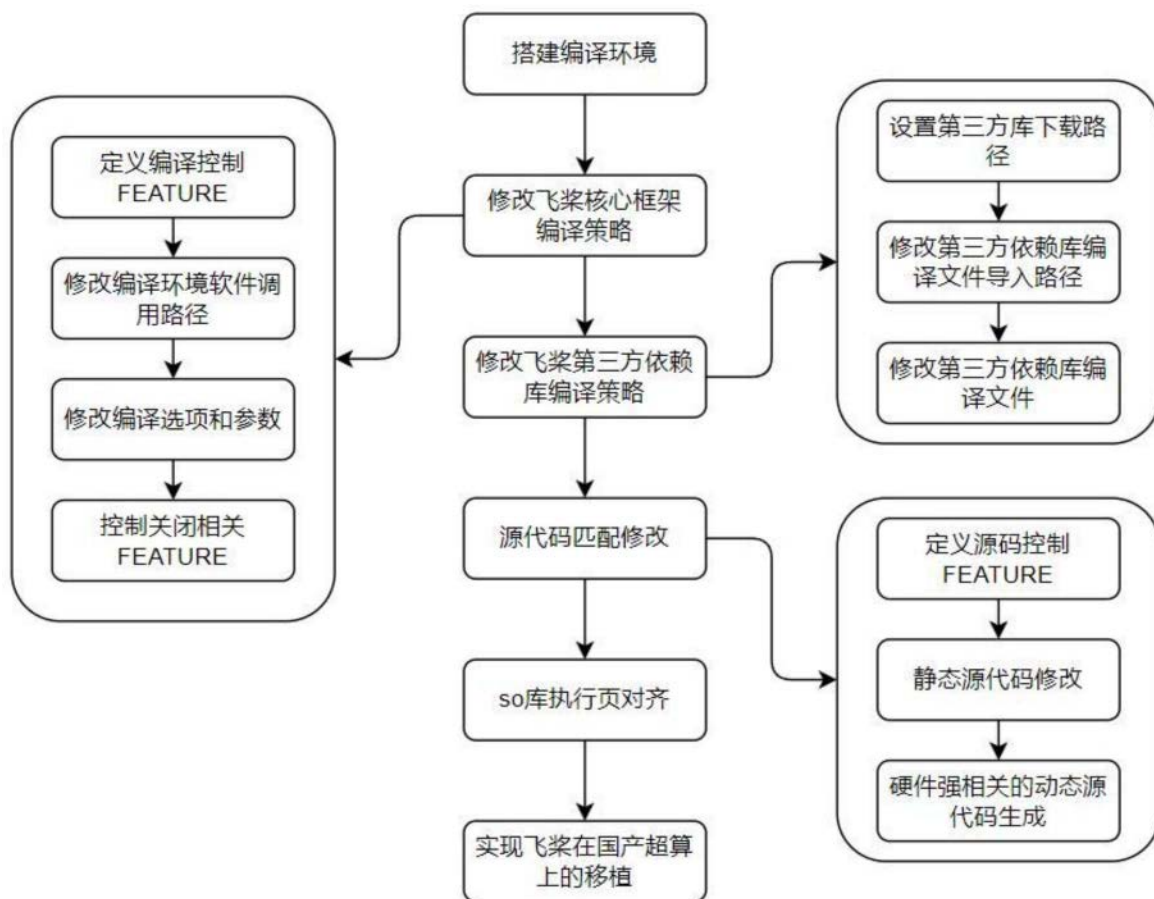


图4

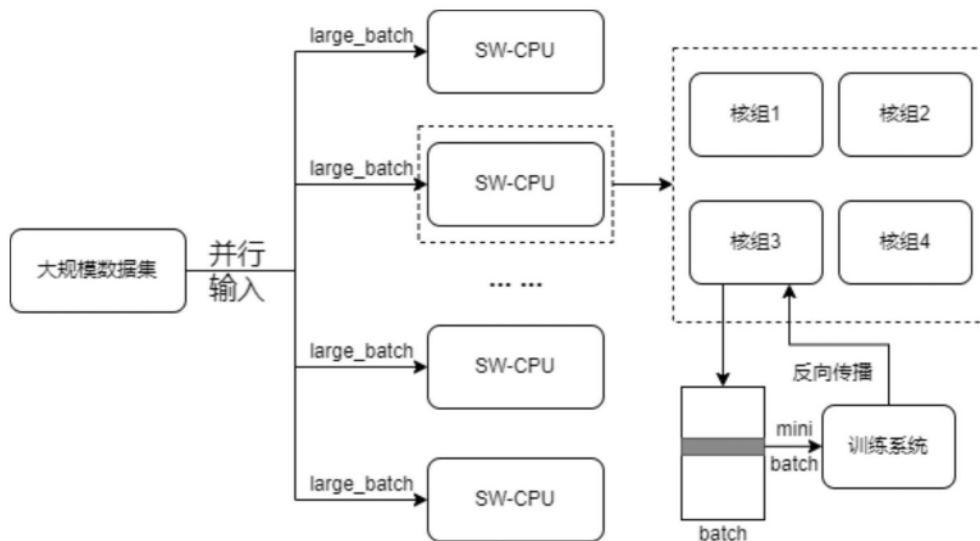


图5

```

[swbd@sw_hpc_43 test_paddle]$ ./bsub.sh
Job <3552148> has been submitted to queue <q_sw_expr>
waiting for dispatch ...
dispatching ...
/usr/sw/swpython/lib/python3.6/site-packages/setuptools-41.0.1-py3.6.egg/setuptools/depends.py:2: DeprecationWarning:
module is deprecated in favour of importlib; see the module's documentation for alternative uses
epoch: 0, batch_id: 100, loss is: [0.3341308], acc is: [0.9]
epoch: 0, batch_id: 200, loss is: [0.29069617], acc is: [0.93]
epoch: 0, batch_id: 300, loss is: [0.12894237], acc is: [0.96]
epoch: 0, batch_id: 400, loss is: [0.15060586], acc is: [0.93]
epoch: 0, batch_id: 500, loss is: [0.13913937], acc is: [0.95]
epoch: 0, batch_id: 600, loss is: [0.08127226], acc is: [0.97]
epoch: 1, batch_id: 100, loss is: [0.03714379], acc is: [0.99]
epoch: 1, batch_id: 200, loss is: [0.12305287], acc is: [0.96]
epoch: 1, batch_id: 300, loss is: [0.16891691], acc is: [0.94]
epoch: 1, batch_id: 400, loss is: [0.15466933], acc is: [0.96]
epoch: 1, batch_id: 500, loss is: [0.05015212], acc is: [0.99]
epoch: 1, batch_id: 600, loss is: [0.14471988], acc is: [0.95]
epoch: 2, batch_id: 100, loss is: [0.08370167], acc is: [0.97]
epoch: 2, batch_id: 200, loss is: [0.13980885], acc is: [0.96]
epoch: 2, batch_id: 300, loss is: [0.06141585], acc is: [0.97]

```

图6

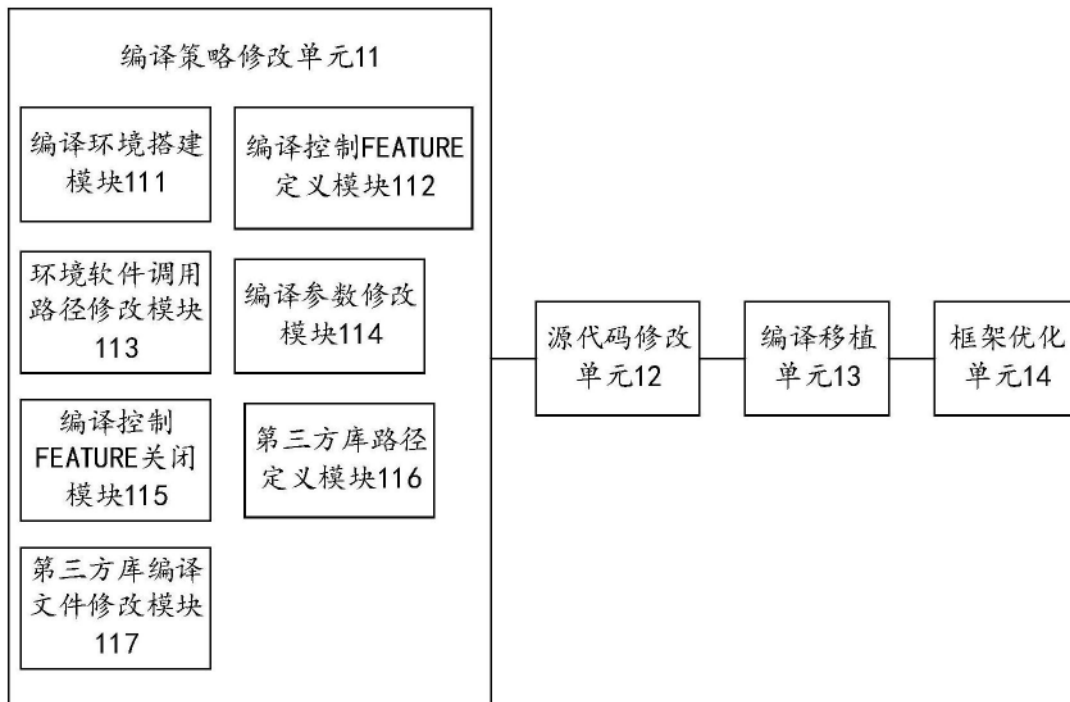


图7

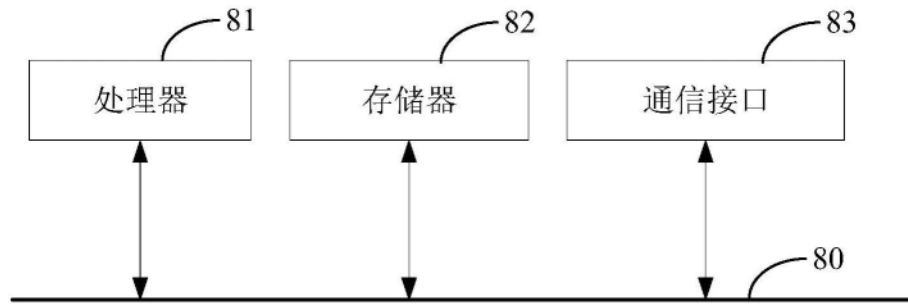


图8