

CPU-GPU 协同高性能卫星数传预处理方法

张鑫宇^{1,2}, 杨甲森¹, 徐 聪^{1,2}, 陈志敏¹, 智 佳¹, 陈 托¹

(1. 中国科学院国家空间科学中心, 北京 100190; 2. 中国科学院大学 计算机科学与技术学院, 北京 100049)

摘 要: 空间数据系统咨询委员会(CCSDS)协议的分层特征对数传预处理的完全并行提出挑战, 虚拟信道、应用过程的多路复用为并行处理提供契机。本文面向高性能数传预处理需求, 在分析处理性能瓶颈的基础上, 提出一种层间流程中央处理器(CPU)控制、层内瓶颈步骤 GPU 加速的协同处理新方法。以高级在轨系统(AOS)帧循环冗余校验(CRC)、工程参数提取与物理量转换算法为研究对象, 对图形处理器(GPU)线程分配、CPU-GPU 协同任务划分进行设计。实验结果表明: 方法可实现 CRC 校验 $11.449\ 6\ \text{GB}\cdot\text{s}^{-1}$ 、工程参数提取与物理量转换 $0.902\ 4\ \text{GB}\cdot\text{s}^{-1}$ 的处理速率, 性能较传统 CPU 架构提升显著。

关键词: 卫星; 并行处理; 数传数据预处理; 中央处理器(GPU); 统一计算设备架构(CUDA)

中图分类号: TP 391 文献标志码: A

DOI: 10.19328/j.cnki.2096-8655.2023.04.005

High-Performance Pre-Processing Method for Transmission Data Based on CPU-GPU Collaboration

ZHANG Xinyu^{1,2}, YANG Jiasen¹, XU Cong^{1,2}, CHEN Zhimin¹, ZHI Jia¹, CHEN Tuo¹

(1. National Space Science Center, Chinese Academy of Sciences, Beijing 100190, China;

2. School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: The pre-processing of transmission data cannot be fully parallelized due to the layering characteristics of the Consultative Committee for Space Data Systems (CCSDS) space communication protocols. Meanwhile, the multiplexing mechanisms of virtual channels and application processes provide an opportunity for parallel processing. In this paper, based on the analysis of processing performance bottlenecks, a high-performance pre-processing method for transmission data based on the central processing unit (CPU) -graphics processing unit (GPU) collaboration is proposed. In this method, the CPU controls the inter-layer flow and intra-layer bottleneck steps accelerated by the GPU. The GPU thread allocation and the assignment of responsibility between the CPU and the GPU are designed to accelerate the cyclic redundancy check (CRC) of the advanced orbiting system (AOS) frames, parameter extraction, and parameter transformation. The experimental results show that the method proposed in this paper can achieve a processing rate of $11.449\ 6\ \text{GB}\cdot\text{s}^{-1}$ for the CRC calibration and a processing rate of $0.902\ 4\ \text{GB}\cdot\text{s}^{-1}$ for the engineering parameter extraction and transformation. It offers significant performance improvements over traditional CPU architectures.

Key words: satellite; parallel processing; transmission data pre-processing; graphics processing unit (GPU); compute unified device architecture (CUDA)

0 引言

数传预处理是地面应用系统生产数据产品、获取探测成果的前提和基础^[1], 其处理内容包括传输

帧同步、帧校验、分虚拟信道、源包同步、分应用过程、工程遥测参数提取与物理量转换、科学数据解压缩等。高分辨率、高灵敏度、多频段覆盖的载荷

收稿日期: 2023-02-28; 修回日期: 2023-06-14

基金项目: 空间科学先导专项科学卫星任务运控技术(XDA15040100)

作者简介: 张鑫宇(1998—), 男, 硕士, 主要研究方向为有效载荷综合测试、空间数据处理。

通信作者: 杨甲森(1979—), 男, 博士, 研究员, 主要研究方向为有效载荷综合测试、空间数据处理。

探测趋势下,Gbps级数传预处理需求日益迫切。

我国航天器数传大多采用的空间数据系统咨询委员会(Consultative Committee for Space Data Systems, CCSDS)相关协议具有典型的分层特征^[2],层间数据相互依赖,层内不同虚拟信道、应用过程数据相对独立。因此,目前国内外学者大多采用中央处理器(Central Processing Unit, CPU)单机、集群等同构平台,以虚拟信道、应用过程为任务划分依据,并行提高处理性能。包括基于消息传递接口(Message-Passing Interface, MPI)^[3]、基于 MPI+OpenMP^[4]在分布式集群上实现卫星遥感数据处理;基于 Storm 流式计算框架实现遥感卫星快视处理^[5]、空间科学卫星数据处理^[6-7]。CPU 适用于处理复杂控制逻辑、较小数据规模问题^[8],限制了上述方法部分处理步骤性能提升。图形处理器(Graphics Processing Unit, GPU)以其 TFLOPS 量级的计算能力,在信号处理^[9-10]、译码^[11-12]、遥感图像处理^[13-18]、卫星轨道计算^[19-20]等并行程度较大的数据密集型领域应用中取得了良好的加速效果。

本文面向高性能数传预处理需求,在分析处理性能瓶颈的基础上,针对预处理步骤层间依赖、各虚拟信道保序并行的特点,提出一种层间流程 CPU 控制、层内瓶颈步骤 GPU 加速的协同处理新方法。对数传数据预处理的 CPU-GPU 协同任务划分、GPU 线程分配进行设计,实现基于 GPU 的帧校验、工程参数提取与物理量转换的加速,为数传预处理性能的提升提供一种新的解决途径。

1 数传数据预处理分析

1.1 数传预处理内容说明

如图 1 所示,在地面接收分系统完成译码后,数传预处理完成高级在轨系统(Advanced Orbiting Systems, AOS)帧处理、源包处理,将结果分发至后续环节。涉及到 CCSDS 协议包括空间包协议^[21]、AOS 空间数据链路协议^[22]、遥测同步和信道编码协议^[23]等,其中 AOS 传输帧、空间包结构见表 1 和表 2。

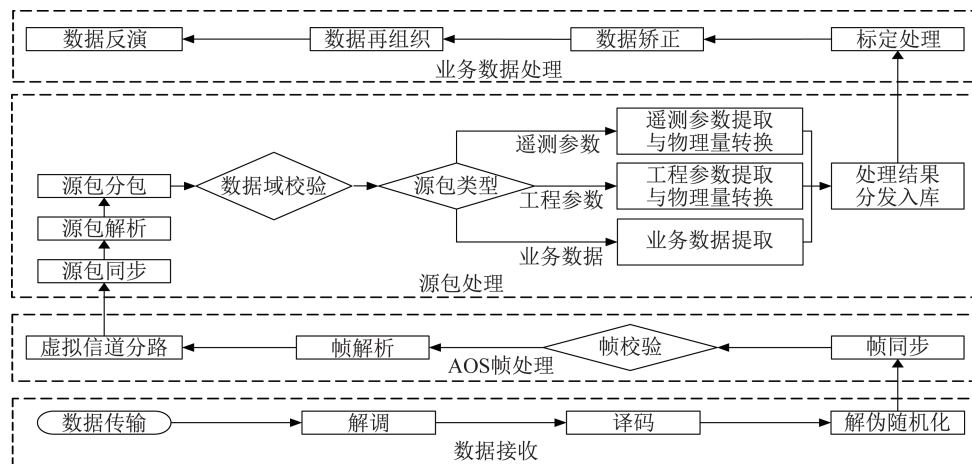


图 1 地面端数据处理的一般过程

Fig. 1 General process of ground data processing

表 1 AOS 传输帧格式

Tab. 1 AOS transfer frame format

传输帧主导头									传输帧插入域 (可选)	传输帧数据域	帧尾(可选)	
主信道标识符 (MCID)		虚拟信道标识符 (VCID)	虚拟信道帧计数	信号域				帧头差错控制 (可选)	特定任务数据 (可选)		操作控制域	帧差错控制域
传输帧版本号 (TFVN)	航天器标识符 (SCID)			回放标志	帧计数周期使用标志	保留域	帧计数周期				可选	可选
2 bits	8 bits	6 bits	24 bits	1 bit	1 bit	2 bits	4 bits	16 bits	变长	变长	32 bits	16 bits

表 2 空间包结构

Tab. 2 Structural components of the space packet

包导头							包数据域	
包标识				包序列控制		包长度	包副导头(可选)	用户数据域
版本号	包类型	副导头标志	应用过程标识符(APID)	序列标志	序列计数			
3 bits	1 bit	1 bit	11 bits	2 bits	14 bits	6 bits	变长	变长

1.2 性能处理瓶颈及可并行性分析

利用文件回放模拟数据输入,测试 CPU 单线程模式下预处理各操作速率见表 3。

表 3 CPU 单线程热点测试

Tab. 3 Hotpot test of the CPU single-threaded program

操作	处理内容	速率/(GB·s ⁻¹)	可并行性分析说明
同步	从码流数据中查询同步码,确定各传输帧起始位置	0.464	低,数据存在依赖
解伪随机化	使用伪随机序列与传输帧除同步码外的每一位二进制数据进行异或操作,恢复出未加扰数据	0.204	高,可各传输帧并行
帧/源包解析	解析帧/包导头中的各字段,与配置比对固定内容字段是否正确	1.218	高,可各传输帧/源包并行
分路分包	提取帧/包数据域数据,按 VCID/APIID 写入缓存	0.723	低,数据有保序要求
帧校验	对 AOS 帧除差错控制域外字段做循环冗余校验(Cyclic Redundancy Check, CRC)校验	0.215	高,可各传输帧并行
参数提取与物理量转换	提取各参数源码并按配置方法进行数学转换	0.004	高,可各源包并行

基于 CCSDS 协议的分层模型和多路复用机制,预处理过程中数据结构变化如图 2 所示。处理存在如下特点:

- 1) 分层模型使各层数据的处理可任务并行;
- 2) 上层协议数据处理依赖下层输出,不同层级数据间存在关联;

3) 多路复用机制使同一物理信道中分属不同虚拟信道数据帧间、同一虚拟信道分属不同应用过程的源包间无依赖关系,可完全并行;

4) 存在保序约束,须保持各虚拟信道、各应用过程内数据的顺序,发挥计数字段检测数据丢失、重复、失序的作用,可保证源包及应用过程数据正确拼接。

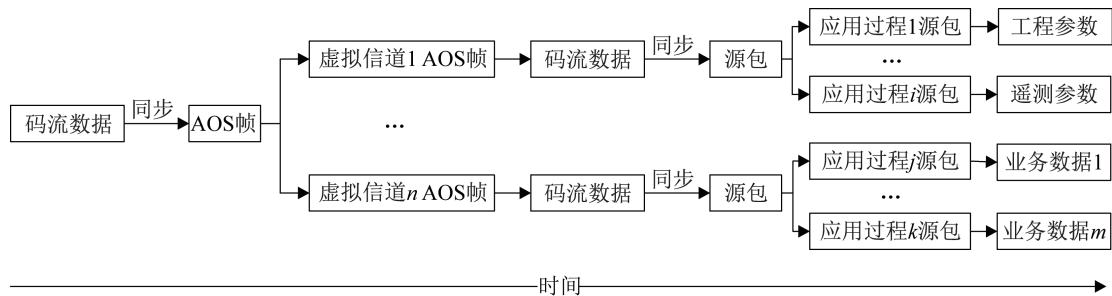


图 2 预处理中的数据结构变化

Fig. 2 Data structure change in pre-processing

2 CPU-GPU 协同数传数据预处理方法

依据预处理步骤层间依赖、各虚拟信道保序并

行的特点,设计 AOS 帧、源包处理步骤的 CPU、GPU 任务分划,并基于 GPU 统一计算设备架构(Com-

pute Unified Device Architecture, CUDA), 实现帧校验、工程参数提取与物理量转换步骤的加速。

2.1 CPU-GPU 协同任务划分

GPU 含有上千个 CUDA 核心, 一次可支持几万个线程并发。GPU 的寄存器带宽为 $\text{TB} \cdot \text{s}^{-1}$ 量级, 远高于 $\text{GB} \cdot \text{s}^{-1}$ 量级的 GPU 显存带宽。CPU 与 GPU 间传输数据的高速串行计算机扩展总线 (Peripheral Component Interconnect Express, PCIe) 带宽远又小于 GPU 显存带宽。因此使用 GPU 获得良好加速效果, CPU、GPU 任务分划要遵循以下原则^[24]:

- 1) 增大核函数并行规模, 避免核心空闲;
- 2) 提高 GPU 中核函数算数强度, 降低 GPU 内存读写占比;
- 3) 降低 CPU-GPU 间数据传输量。

2.1.1 AOS 帧处理流程设计

分路分包前码速率较高, 而帧校验又为计算量较大的瓶颈步骤; 各 AOS 帧独立校验, 帧间无依赖关系; 高码速率同时降低了等待数据积累到较大规模供 GPU 处理的时延, 基于上述考虑, 如图 3 所示, 设计 GPU 加速帧校验与解析。使用 CPU 对解扰后的码流数据同步, 得到 AOS 帧写入帧缓存, 经过 PCIe 总线传输至 GPU 全局内存。GPU 处理完毕后不再向 CPU 端传回完整 AOS 帧, 而是传回 8 bit/帧的后续处理必须信息, 数据传输量降低 99.1%。传回信息包括: 帧格式符合性、VCID、虚拟信道帧计数、数据域有效数据结束地址。CPU 端异常帧报警器在检测到帧格式不符合时, 重新解析该帧, 生成异常帧报告; 帧数据域提取处理器提取数据, 按 VCID 写入各源包码流数据环形队列存储器, 同步处理后得到源包 (空间包)。

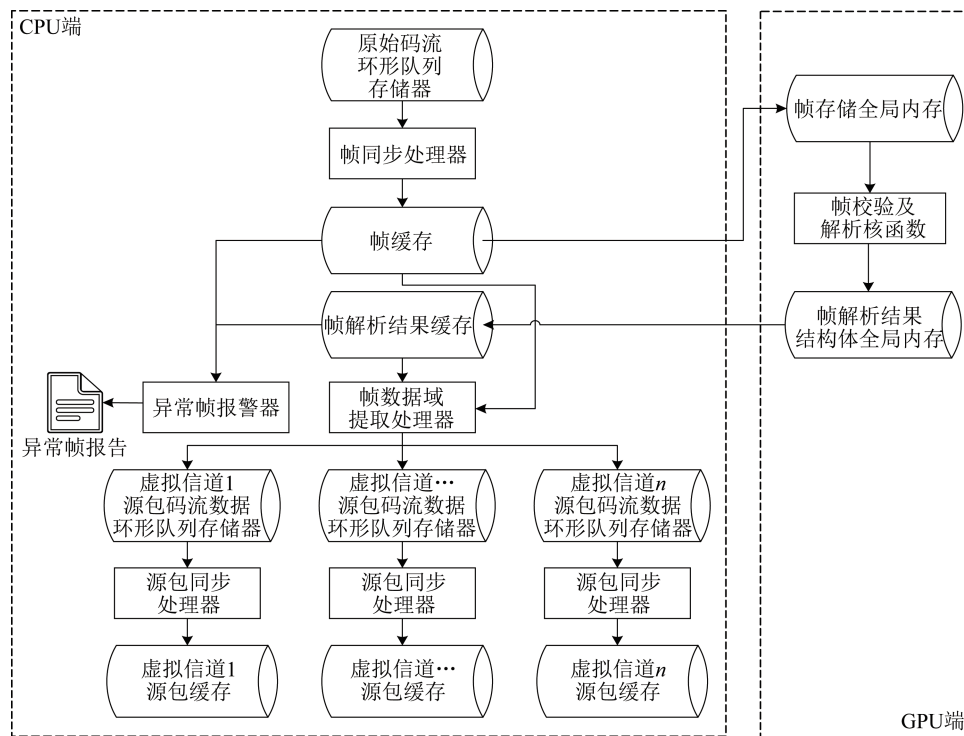


图 3 CPU-GPU 协同的 AOS 帧处理流程

Fig. 3 Processing flow of AOS frame parsing based on the CPU-GPU collaboration

2.1.2 源包处理流程设计

源包按装载的数据类型, 可分为业务数据源包、遥测参数源包、工程参数源包。空间科学卫星的科学数据、遥感卫星的图像数据等业务数据, 数据量大、处理方法复杂、不同业务处理差异大。遥测数据是与航天器工作状态有关的测量数据 (如电

压、电流、温度、压力等), 反映航天器的工作过程、工作结果和健康状态^[25], 实时性要求高, 数据量较小。工程参数主要是有效载荷的工作状态数据, 相比遥测参数, 种类更多、数据量更大。

根据不同类型的源包特点, 设计源包处理任务流程如图 4 所示。CPU 解析源包后生成异常源包

报告,根据 APID 将格式符合的源包写入相应的环形队列存储器。业务数据在预处理阶段只做拼接处理,由 CPU 完成。参数配置信息处理器读取含有参数位置、源数据长度、物理量转换方法的配置文

件,计算掩码,在程序启动时向 GPU 全局内存传输一次。遥测参数数据量较小,为降低时延,由 CPU 完成提取与物理量转换。工程参数由 GPU 完成提取与物理量转换。

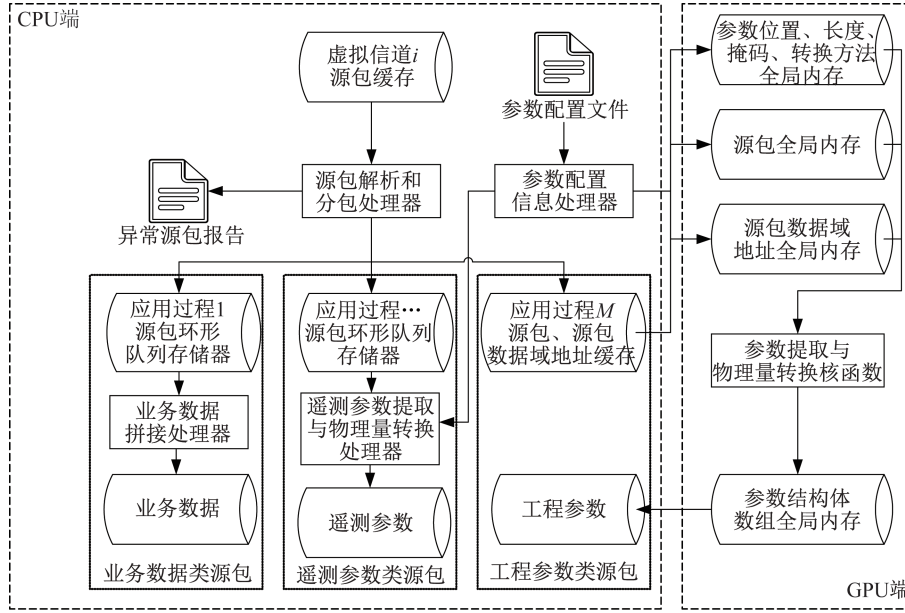


图 4 CPU-GPU 协同的源包处理流程

Fig. 4 Processing flow of space packet parsing based on the CPU-GPU collaboration

2.2 基于 GPU 的并行加速实现

2.2.1 帧校验的 CUDA 加速

AOS 协议的差错控制使用生成多项式为 $C(x) = x^{16} + x^{12} + x^5 + 1$ 的 CRC 校验, CRC 校验的基本思想是接收端的数据无误码则能被给定的生成多项式 $C(x)$ 整除。CRC 校验以帧为基本单位,各帧之间不影响,设计每个 GPU 线程校验一帧。从主机端缓存每次读取 N 个长度均为 L 字节的 AOS 帧至 GPU 全局内存中线性存储。将网格内的线程块和线程块内的线程均组织成一维,线程块数量为 N_b ;每个网格中线程数量 N_g 为

$$N_g = \begin{cases} N/N_b, N \% N_b = 0 \\ N/N_b + 1, N \% N_b \neq 0 \end{cases} \quad (1)$$

每个线程在核函数中的唯一身份标识 i_{dx} 为

$$i_{dx} = T_x + P_b \cdot D_b \quad (2)$$

式中: $T_x \in [0, D_b - 1]$ 为各线程在其线程块中的索引。

第 i_{dx} 个线程处理第 i 个传输帧,该帧的数据在全局内存地址为: $[i_{dx} \cdot L, (i_{dx} + 1) \cdot L - 1]$,每个线

程完成校验后,解析帧主导头各字段,提取所需信息写入帧解析结果结构体。所有线程计算完毕,使用 `cudaMemcpy()` 函数将帧解析结果结构体数组传输至 CPU 端内存。

2.2.2 工程参数提取与物理量转换的 CUDA 加速

参数在源包中以二进制信号形式存在,需要将参数提取、编码、转换后,才能直观地用于载荷状态监视、参数判读等任务,可分为以下 3 个步骤:

步骤一 参数提取。提取源包数据域码流中各参数的二进制信号源码。

参数位置由参数开始的字节地址 A_b 、开始字节的位地址 A_b 、比特长度 L_b 唯一确定。某参数在源包中所占字节长度 L_b 为

$$L_b = \left\lceil \frac{7 - A_b + L_b}{8} \right\rceil \quad (3)$$

将帧中第 A_b 至 $A_b + L_b - 1$ 字节数据与位掩码做按位与运算并位移后得到参数源码。

步骤二 参数编码。将各参数的二进制信号源码编码为实际数字或字符串。

步骤三 物理量转换。参数经某种方法的运

算后,转换为直观表达真实物理含义的值。常用的物理量转换方法有源码显示、进制转换、含义映射、多项式变换、热敏电阻分压值转换为温度等。

从CPU端缓存每次读取特定种类长度为 L 字节,含 M 个参数的工程参数源包 N 个,每个参数的处理结果使用 R 字节保存。将网格内的线程块和线程块内的线程均组织成一维,线程块数量为 N_b 。设计了每个线程处理一个参数、每个线程处理一个源包2种线程分配方案,通过实验分析2种方案的特性。

方案一 每个线程处理一个参数。

每个网格中 x 方向线程数量 N_g 为

$$N_g = \begin{cases} N*M/N_b, (N*M)\%N_b = 0 \\ N*M/N_b + 1, (N*M)\%N_b \neq 0 \end{cases} \quad (4)$$

第 i_{dx} 个线程处理第 $\lceil i_{dx}/M \rceil$ 个源包的第 $(i_{dx}\%M)$ 个参数,将结果写入GPU全局内存,内存

地址为 $[i_{dx}*R, (i_{dx}+1)*R-1]$ 。

方案二 每个线程处理一个源包。

每个网格中 x 方向线程数量 N_g 计算方法与公式(1)相同,第 i_{dx} 个线程处理第 i_{dx} 个源包的所有参数,将结果写入GPU全局内存,内存地址为 $[i_{dx}*M*R, (i_{dx}+1)*M*R-1]$ 。

2.2.3 基于CUDA流并发执行核函数和数据传输

CUDA流是从主机发出在GPU中执行的一系列异步CUDA操作,不同CUDA流中的操作可能并发或交错执行^[14]。由于数据传输、计算分别通过GPU的读写单元和流处理器进行,如图5所示,主机向一个流发出数据传输命令后可立即获得控制权,到另一个流调用核函数,从而重叠数据传输与核函数执行^[15]。在CUDA操作完全并行执行的理想情况下,数据传输时间可完全隐藏,随着流数量增加,加速比趋近3。

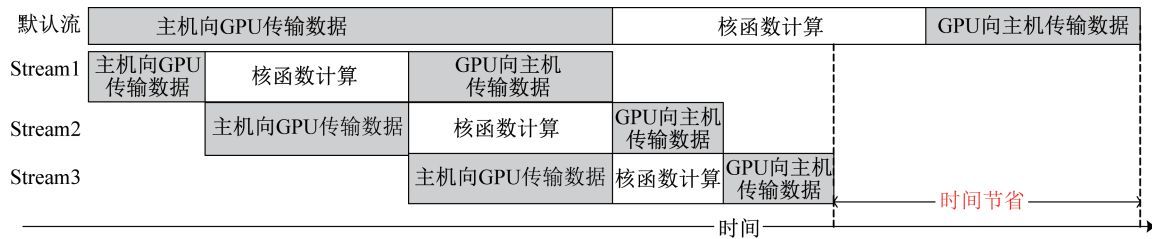


图5 CUDA流并发示意

Fig. 5 CUDA stream concurrency

CUDA流实现异步数据传输,需要在程序运行期间将主机端数据缓存定义为不可分页内存,保持内存物理地址不变。设创建 N_s 个CUDA流,将每次从缓存中读取的 N 个AOS帧或源包划分 N_s 个子集,平均分配到各个流中,使用cudaMemcpyAsync()函数异步传输数据,使用循环调度各个流的数据传输和计算。

3 实验验证与分析

3.1 实验环境

CPU使用AMD Ryzen 5 2600X,基频3.6 GHz。GPU使用NVIDIA GeForce RTX 2080ti,显卡为图灵架构,有68个流多处理器,4 352个CUDA核心,11 GB GDDR6显存,显存带宽 $616 \text{ GB}\cdot\text{s}^{-1}$,单精度浮点数运算峰值 13 TFLOPS ,CUDA版本10.1。CPU与GPU之间使用PCIe3.0 x16总线连接,理论带宽 $15.754 \text{ GB}\cdot\text{s}^{-1}$ 。内存使用48GB DDR4双通道

内存。硬盘使用512 GB NVMe固态硬盘。

由于不同型号CPU物理核心数的差距,实践中评价加速效果常使用CPU单核与GPU比较^[9]。

3.2 帧校验实验结果与分析

传输帧帧长898字节,前4字节为同步码,帧差错控制域2字节。 N_b 设为256。如图6所示,使用单个CUDA流,一次处理 2^{15} 个帧时达到最高处理速率 $8.0536 \text{ GB}\cdot\text{s}^{-1}$,加速比38.003;使用16个CUDA流,一次处理 2^{19} 个帧时达到最高处理速率 $11.4496 \text{ GB}\cdot\text{s}^{-1}$,加速比53.866。CPU单线程方式,最高处理速率 $0.2147 \text{ GB}\cdot\text{s}^{-1}$,串行计算性能基本与帧数量无关。

如图7所示,一次处理的帧数量少时,GPU没有满负荷工作,有空闲的计算资源,增加一次处理的帧数量,GPU核函数耗时基本不变;当一次处理的帧个数较大时,GPU满负荷工作,GPU核函数的

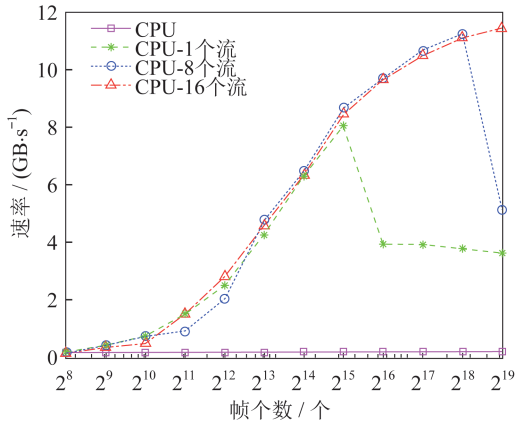


图 6 速率随AOS帧个数的变化-CRC

Fig. 6 Variation of the processing speed with the AOS frame number-CRC

耗时正比于一次处理的帧个数。可根据不同GPU性能,配置单次计算规模,达到最佳性能。

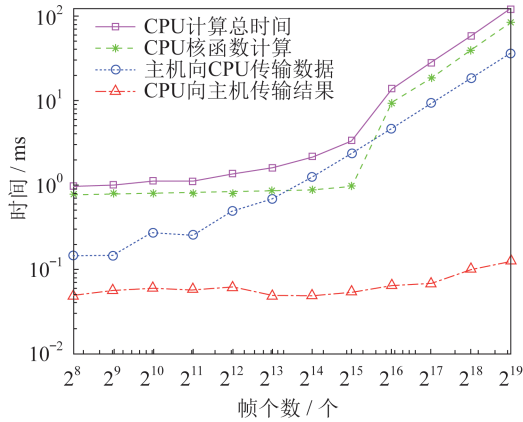


图 7 GPU核函数计算与数据传输耗时-CRC

Fig. 7 Time of the kernel function and data transmission by GPU-CRC

3.3 工程参数提取与物理量转换实验结果与分析

工程参数源包包长 390 字节,含 467 个参数,每个参数转换结果 8 字节保存。 N_b 设为 256。如图 8 所示,CPU 最高速率 $0.0044 \text{ GB}\cdot\text{s}^{-1}$ 。使用单个 CUDA 流,最高速率 $0.506 \text{ GB}\cdot\text{s}^{-1}$,加速比 115.98;使用 16 个 CUDA 流,最高速率可达 $0.9024 \text{ GB}\cdot\text{s}^{-1}$,加速比 211.021。

如图 9 和图 10 所示,一次处理的源包个数小于 2^{15} 时,每个线程处理一个源包的方案并行规模较小,速率比每个线程处理一个参数的方案低;源包个数大于 2^{15} 时,两方案均能使 GPU 满负荷工作,每个线程处理一个源包的算数强度更大,速率更高。

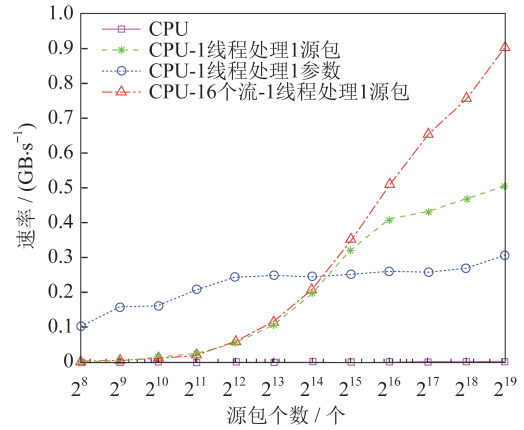


图 8 速率随源包个数的变化

Fig. 8 Variation of the processing speed with the packet number

在实际应用中,可根据数据的时延性要求,配置一次处理的源包个数,进而确定线程分配方案。

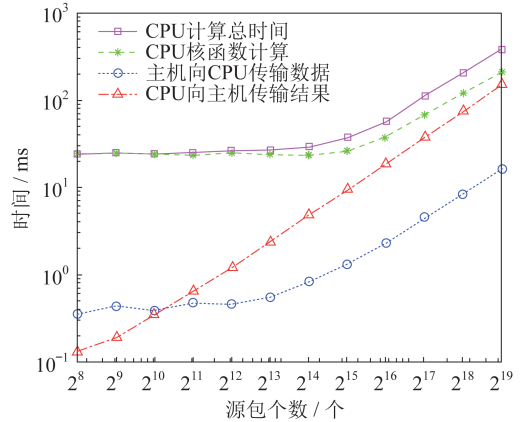


图 9 GPU核函数计算与数据传输耗时-1线程处理1源包

Fig. 9 Time of the kernel function and data transmission by GPU-1 packet for 1 thread

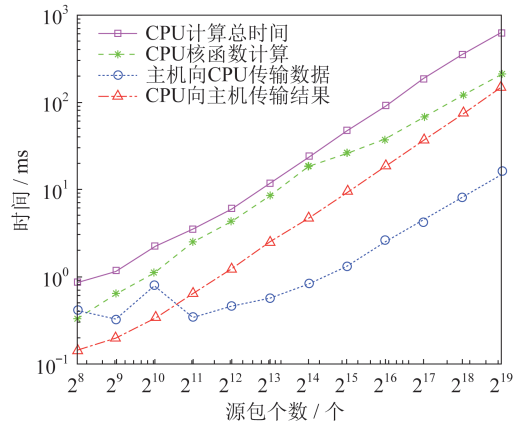


图 10 GPU核函数计算与数据传输耗时-1线程处理1参数

Fig. 10 Time of the kernel function and data transmission by GPU-1 parameter for 1 thread

4 结束语

数传数据预处理是卫星数据处理的前置必须环节,本文提出的CPU-GPU协同的卫星数传数据预处理方法,利用GPU并行计算加速处理过程的帧校验、参数提取与物理量转换等耗时瓶颈步骤,实验结果表明,该方法可满足Gbps的卫星数传数据预处理需求。后续可开展的工作如下。

1) 增加处理深度,引入基于GPU的数据解压缩、科学数据快视、数据判读,进一步加速数据处理过程。

2) 研究适用于数传数据处理的多CPU、多GPU场景中有硬件和任务优先级约束下的多目标优化调度算法。

参考文献

- [1] 杨甲森,孟新,王春梅.多星多任务数传数据实时处理系统设计[J].航天器工程,2016,25(4):60-66.
- [2] CCSDS. Overview of space communications protocols: 130.0-G-3[S]. CCSDS, 2014.
- [3] 刘莉.基于MPI的卫星遥感数据实时处理平台设计[J].航天器工程,2013,22(3):130-134.
- [4] 张人愉,李景山.通用遥感卫星基带数据二级并行处理算法设计与实现[J].遥感信息,2019,34(3):54-61.
- [5] 宋晓,孙小涓,胡玉新,等.基于流式计算的遥感卫星数据快视处理方法[J].计算机工程与应用,2019,55(10):77-82.
- [6] 孙小涓,石涛,李冰,等.空间科学卫星数据快速处理方法[J].计算机工程与科学,2018,40(8):1351-1357.
- [7] 孙小涓,石涛,胡玉新,等.基于流式计算的空间科学卫星数据实时处理[J].计算机应用,2019,39(6):1563-1568.
- [8] 邬江兴,祁晓峰,高彦钊.异构计算并行编程模型综述[J].上海航天(中英文),2021,38(4):1-11.
- [9] 龚昊,刘莹,冯建周,等.基于GPU加速的脉冲多普勒雷达信号处理[J].计算机工程与科学,2021,43(7):1141-1149.
- [10] 赵志欣,翁涛,张凯凯,等.基于GPU的OFDM波形外辐射源雷达信号处理研究[J].现代电子技术,2021,44(17):6-11.
- [11] 侯毅,刘荣科,彭皓,等.适用于空间通信的LDPC码GPU高速译码架构[J].航空学报,2017,38(1):236-245.
- [12] 孔飞跃,蒋学芹,万雪芬,等.基于GPU的LDPC增强最大似然译码器并行实现[J].计算机工程,2020,46(5):207-215.
- [13] 方留杨.CPU/GPU协同的光学卫星遥感数据高性能处理方法研究[D].湖北:武汉大学,2015.
- [14] 陈凯,曹云刚,杨秀春,等.基于CPU-GPU异构混合编程的遥感数据时空融合[J].地理信息世界,2019,26(6):6-13.
- [15] 赵晓晨,吴皓楠,李林宜,等.面向汛旱情监测的遥感影像GPU并行处理算法[J].自然资源遥感,2021,33(3):107-113.
- [16] 程富雕,王密,张致齐,等.对吉林一号光学卫星的应急快速处理方法研究[J].测绘地理信息,2017,42(4):47-51.
- [17] 董敏,阎镇.基于CUDA的航天遥感图像实时快视系统的研究[J].计算机技术与发展,2014,24(6):32-35,39.
- [18] 李盛阳,刘志文,张万峰,等.天宫二号应用任务地面数据处理与服务系统[J].中国科学:技术科学,2020,50(8):1066-1080.
- [19] 计天阳.基于GPU加速的低轨卫星星座覆盖性能计算与分析[D].哈尔滨:哈尔滨工业大学,2018.
- [20] 孔繁泽,叶东,柳子然,等.卫星轨道递推的GPU集成式并行加速方法[J].哈尔滨工业大学学报,2021,53(6):13-20.
- [21] CCSDS. Space packet protocol: 133.0-B-2 [S]. CCSDS, 2020.
- [22] CCSDS. AOS space data link protocol: 732.0-B-3[S]. CCSDS, 2015.
- [23] CCSDS. TM synchronization and channel coding: 131.0-B-3[S]. CCSDS, 2017.
- [24] 樊哲勇.CUDA编程:基础与实践[M].北京:清华大学出版社,2020:46-55.
- [25] 赵和平,何熊文,刘崇华,等.空间数据系统[M].北京:北京理工大学出版社,2018:4-5.