(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105224506 B
(45)授权公告日 2018.02.13

(21)申请号 201510725866.3

(22)申请日 2015.10.29

(65)同一申请的已公布的文献号
申请公布号 CN 105224506 A

(43)申请公布日 2016.01.06

(73)专利权人 北京大学
地址 100871 北京市海淀区颐和园路5号

(72)发明人 陈一峯 崔翔 王韬 严磊

(74)专利代理机构 北京万象新悦知识产权代理事务所(普通合伙) 11360
代理人 张肖琪

(51)Int.Cl.
G06F 17/14(2006.01)

审查员 葛晓倩

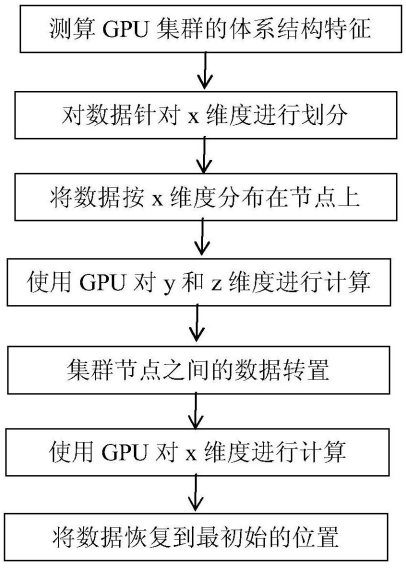
权利要求书1页 说明书8页 附图2页

(54)发明名称

一种用于GPU异构集群的高性能FFT方法

(57)摘要

本发明公布了一种用于GPU异构集群的高性能FFT方法，将数据进行维度划分并表示为数组数据，按照划分维度进行分割后分布到GPU异构集群中与划分维度相同数目的节点上，再进行FFT计算，包括：获得GPU集群的节点数目、节点上GPU数目、节点内存容量、GPU设备存储器容量信息；将数据采用数组维度表示为数组数据；针对x维度进行维度划分；按照x维度分割数据，分布在多个节点上；将y和z维度方向的数据进行FFT计算；进行集群节点之间的数据转置；进行x维度的FFT计算。本发明是针对GPU异构集群的多层存储器体系结构特征的FFT方法，具有良好的性能和可延展性。



CN 105224506 B

1.一种用于GPU异构集群的高性能FFT方法，所述方法将待处理数据进行维度划分并表示为数组数据，对数组数据按照划分维度进行分割后将数据分布到GPU异构集群中与划分维度相同数目的节点上，再进行FFT计算；具体包括如下步骤：

1)测试获得所使用GPU集群的体系结构特征，包括节点数目、节点上GPU数目、节点内存容量、GPU设备存储器容量信息；设定d为GPU异构集群的节点数目；

2)将待处理数据采用数组维度表示方法表示为数组数据；针对x维度进行维度划分，x维度的大小设为a，记为$\frac{x}{[a]}\frac{y}{[b]}\frac{z}{[c]}$，其中，x、y和z为维度的名称，a、b和c为相应维度的大小；

3)将步骤2)所述数组数据按照x维度分割为d份，数据分布在a个节点上；每一个节点的数据为(a/d)*b*c个复数，此时数据记为$\frac{\hat{x}_1}{[d]}\frac{x_0}{[a/d]}\frac{y}{[b]}\frac{z}{[c]}$，其中，x维度分裂为$\hat{x}_1$和$x_0$两个维度，维度大小分别为d和a/d；

4)每个节点上的GPU将y维度方向和z维度方向的数据进行FFT计算；具体是进行x_0维度个FFT计算，即进行a/d个2维的FFT计算；

5)在进行x维度的FFT计算之前，进行集群节点之间的数据转置，将y维度的数据做进一步的划分，得到$\frac{\hat{x}_1}{[d]}\frac{x_0}{[a/d]}\frac{y_1}{[d]}\frac{y_0}{[b/d]}\frac{z}{[c]}$，在集群上直接进行通信，将y维度和x维度进行交换，数据分布变换为$\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\left[\frac{x_1}{[d]}\frac{x_0}{[a/d]}\right]\frac{z}{[c]}$，得到$\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\frac{x}{[a]}\frac{z}{[c]}$，数据按照$y_1$维度分布在集群节点上，x维度方向的数据存储在各个节点上；

6)各个节点分为y_0维度大小b/d次，将a*c的数据拷贝到GPU设备存储器上，并进行二维数据转置，使得x维度的数据完全连续，得到$\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\frac{z}{[c]}\frac{x}{[a]}$；

7)进行x维度的FFT计算；

8)反方向重复整个数据移动过程，回溯整个数据转置和传输的过程，将数据恢复为初始的分布状态，最终将数据恢复到最初始的位置，即$\frac{x}{[a]}\frac{y}{[b]}\frac{z}{[c]}$。

2.如权利要求1所述用于GPU异构集群的高性能FFT方法，其特征是，所述FFT计算具体通过调用FFT数学库函数CUFFT实现。

3.如权利要求1所述用于GPU异构集群的高性能FFT方法，其特征是，所述GPU异构集群为天河1A系统。

4.如权利要求1～3任一所述用于GPU异构集群的高性能FFT方法，其特征是，步骤1)所述GPU异构集群的节点数目d为4096；步骤2)所述x、y和z维度相应维度大小a、b和c均为8192。

一种用于GPU异构集群的高性能FFT方法

技术领域

[0001] 本发明涉及计算机高性能计算领域，涉及一种用于GPU异构集群上的高性能FFT(Fast Fourier Transformation，快速傅氏变换)计算方法。

背景技术

[0002] 在异构处理器FFT的研究方面，Nvidia公司有针对单个GPU的数学库CUBLAS和CUFFT，也正在研制针对单机上多个GPU的数学库；很多研究人员进行了单GPU上FFT算法的研究。在发布MIC协处理器的同时，Intel公司也推出了MIC协处理器上的MKL数学库。现有的传统集群上的商业版FFT数学库主要有FFTW和Intel MKL。FFTW是使用最为广泛的FFT库，其基于MPI的3维FFT并没有考虑计算和通信的重叠优化问题；P3DFFT通过对数据进行节点间高维分布以提高3维FFT的延展性，但没有考虑计算和通信的重叠优化问题；2DECOMP&FFT和Kandalla等通过优化多组输入数组间的计算通信重叠问题实现对批量数据3维FFT计算的性能优化；针对一些科学计算问题中需要对一个数据数组进行多次3维FFT计算的应用场景，Song等通过优化一次3维FFT内的计算通信重叠问题来提升性能；Bell等基于UPC和支持异步通信的硬件实现了一次3维FFT内的计算通信重叠优化；Fang等使用特定通信API在多维网络结构中实现了计算通信重叠的3维FFT；Doi等考虑非对称网状网络的拓扑结构，合理调度远距离进程之间的通信，在多核节点上使用不同的线程实现通信和计算的重叠。在HPCC的集群FFT测试中，目前最快的集群FFT是IBM Blue Gene，达到226.10TFlop/s。

[0003] 传统的集群3维FFT可以用维度表示法进行清晰的描述，如图2中的(a)所示，为8×8×8的3维立方体数据，其中，z维度为最密集的，而x维度为最稀疏的，此时，数组可以记为$\frac{x}{[8]}\frac{y}{[8]}\frac{z}{[8]}$。将该数组分布到4个节点的集群上去，一种方式是进行1维的分割，即将该数组按照x维度的方向进行分割，如图2中的(b)所示，此时，该数组记为$\left[\frac{\hat{x}_1}{[4]}\frac{x_0}{[2]}\right]\frac{y}{[8]}\frac{z}{[8]}$，这意味着数据分布在4个节点上，每个节点有2($x_0$维度)个8×8(y和z维度)的正方形数据，进行3维FFT计算的步骤如下：

[0004] 一，在4(x_1维度)个节点上分别并行进行2(x_0维度)次2维8×8的FFT，即将y和z维度方向的1维FFT计算完成；

[0005] 二，需要进行x维度方向的FFT，由于x维度方向的数据分布在不同的节点上，因此需要进行通信。将数据再进行y维度方向的分割，如图2中的(c)所示。此时，该数组记为$\left[\frac{\hat{x}_1}{[4]}\frac{x_0}{[2]}\right]\left[\frac{y_1}{[4]}\frac{y_0}{[2]}\right]\frac{z}{[8]}$。通过进行2次($x_0$维度)的分组AllToAll通信，即可将$x_1$维度和$y_1$维度进行对调，数据的分布变为$\frac{\hat{y}_1}{[4]}\frac{x_0}{[2]}\frac{x_1}{[4]}\frac{y_0}{[2]}\frac{z}{[8]}$；

[0006] 三，数据已经按照y_1维度分布在4个节点上，但x维度的数据在内存中仍不连续。为了使x维度的数据连续以保证cache访问的友好性，也可以在各个节点并行进行本地的数据

转置，将其变为$\frac{\hat{y}_1}{[4]}\frac{y_0}{[2]}\frac{z}{[8]}\left[\frac{x_1}{[4]}\frac{x_0}{[2]}\right]$，即$\frac{\hat{y}_1}{[4]}\frac{y_0}{[2]}\frac{z}{[8]}\frac{x}{[8]}$（此转置需要额外的内存缓冲区空间）；

[0007] 四，在完成了对x维度数据的1维FFT之后，可以将整个步骤反序进行，把数据元素恢复到初始的位置。

[0008] 传统集群3维FFT的另一种分布数据的方法是按照2个维度对数据进行分布，对8×8×8的3维立方体数据按照x维度和y维度进行划分，分布到4×4的节点阵列上，记为$\left[\frac{\hat{x}_1}{[4]}\frac{x_0}{[2]}\right]\left[\frac{\hat{y}_1}{[4]}\frac{y_0}{[2]}\right]\frac{z}{[8]}$，该方法进行3维FFT计算的步骤如下：

[0009] 一，z维度方向的数据都在同一个节点本地，16个节点并行计算2(x_0维度)×2(y_0维度)个z维度方向的1维FFT；

[0010] 二，为了通信的需要，将z维度做进一步的分裂，记为$\left[\frac{\hat{x}_1}{[4]}\frac{x_0}{[2]}\right]\left[\frac{\hat{y}_1}{[4]}\frac{y_0}{[2]}\right]\left[\frac{z_1}{[4]}\frac{z_0}{[2]}\right]$。按照$x_1$维度，将节点阵列分为4组，每组4($y_1$维度)个节点，各组节点内做2($x_0$维度)×2($y_0$维度)次分组AllToAll通信，将$y_1$维度和$z_1$维度转置，得到$\left[\frac{\hat{x}_1}{[4]}\frac{x_0}{[2]}\right]\frac{\hat{z}_1}{[4]}\frac{y_0}{[2]}\frac{y_1}{[4]}\frac{z_0}{[2]}$；

[0011] 三，y维度方向的数据都在同一个节点本地，16个节点并行计算2(x_0维度)×2(z_0维度)个y维度方向的1维FFT；

[0012] 四，按照z_1维度，将节点阵列分为4组，每组4(x_1维度)个节点，各组节点内做2(x_0维度)×2(y_0维度)次AllToAll通信，将x_1维度和y_1维度转置，得到$\frac{\hat{y}_1}{[4]}\frac{x_0}{[2]}\frac{\hat{z}_1}{[4]}\frac{y_0}{[2]}\frac{x_1}{[4]}\frac{z_0}{[2]}$；

[0013] 五，在完成了对x维度数据的1维FFT之后，可以将整个步骤反序进行，把数据元素恢复到初始的位置。

[0014] 在上述计算方法中，第二步和第四步的通信都需要进行四次较小规模的AllToAll通信，在节点具有足够内存容量做缓冲区的情况下，可以通过本地数据转置将通信减少为1个AllToAll通信。

[0015] 综上所述，已有高性能FFT方法针对的是单机或传统集群，到目前，尚未出现基于GPU异构集群的高性能FFT方法。对于GPU异构集群，由于其具有更加复杂的多层存储器结构，传统集群上的FFT算法已经不再适合其体系结构的特征，难以取得更高的性能。

发明内容

[0016] 为了克服上述现有技术的不足，本发明提供一种用于GPU异构集群上的高性能FFT(FastFourier Transformation，快速傅氏变换)计算方法，针对GPU异构集群的多层存储器体系结构特征，通过适应性的数据维度划分和传输转置方法，实现天河1A系统上的大规模3维FFT，该方法得到了良好的性能和可延展性。

[0017] 本发明提供的技术方案是：

[0018] 一种用于GPU异构集群的高性能FFT方法，所述方法将待处理数据进行维度划分并表示为数组数据，对数组数据按照划分维度进行分割，将数据分布到GPU异构集群中与划分

维度相同数目的节点上，再进行FFT计算；图1是本发明方法的流程图，具体包括如下步骤：

[0019] 1)测试所使用GPU集群的体系结构特征，获取包括节点数目、节点上GPU数目、节点内存容量、GPU设备存储器容量等信息；设定d为GPU异构集群的节点数目；

[0020] 2)针对待处理数据，针对x维度进行维度划分，采用数组维度表示方法进行表示，x维度的大小设为a，记为$\frac{x}{[a]}\frac{y}{[b]}\frac{z}{[c]}$，其中，x、y和z为维度的名称，a、b和c为相应维度的大小；该数组表示和传统编程语言中的数组表示相同，只是在各个维度上加上该维度的命名，方便对FFT计算问题的表示；

[0021] 3)将数组数据(一般情况下，数据需为三维结构，但各个维度的大小可以不同)按照x维度分割为d份，数据分布在a个节点上；

[0022] 这样，每一个节点的数据为(a/d)*b*c个复数，此时数据记为$\frac{\hat{x}_1}{[d]}\frac{x_0}{[a/d]}\frac{y}{[b]}\frac{z}{[c]}$，即x维度分裂为$\hat{x}_1$和$x_0$两个维度，维度大小分别为d和a/d；

[0023] 4)每个节点上的GPU将y维度方向和z维度方向的数据进行FFT计算；具体是进行x_0维度个FFT计算，即进行a/d个2维的FFT计算；

[0024] 由于计算的时间与数据传输的时间相比基本上可以忽略，在此直接使用CUFFT调用(CUFFT为Nvidia公司的单GPU上的FFT数学库函数)即可。

[0025] 5)在进行x维度的FFT计算之前，需要进行集群节点之间的数据转置，将y维度的数据做进一步的划分，得到

[0026] $$\frac{\hat{x}_1}{[d]}\frac{x_0}{[a/d]}\frac{y_1}{[d]}\frac{y_0}{[b/d]}\frac{z}{[c]},$$

[0027] 在集群上直接进行通信，将数据分布变换为

[0028] $$\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\left[\frac{x_1}{[d]}\frac{x_0}{[a/d]}\right]\frac{z}{[c]}，即\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\frac{x}{[a]}\frac{z}{[c]},$$

[0029] 这样，y维度和x维度得到交换，数据按照y_1维度分布在集群节点上，x维度方向的数据存储在各个节点之内。

[0030] 6)此时，各个节点可以分b/d(y_0维度)次，将a*c的数据拷贝到GPU设备存储器上，并进行二维数据转置，使得x维度的数据完全连续，得到

[0031] $$\frac{\hat{y}_1}{[d]}\frac{y_0}{[b/d]}\frac{z}{[c]}\frac{x}{[a]};$$

[0032] 7)进行x维度的FFT计算；

[0033] 8)在使用CUFFT进行x维度的1维FFT计算之后，反方向重复整个数据移动过程，即回溯整个数据转置和传输的过程，将数据恢复为初始的分布状态，最终将数据恢复到最初始的位置，即$\frac{x}{[a]}\frac{y}{[b]}\frac{z}{[c]}$。

[0034] 将上述用于GPU异构集群的高性能FFT方法应用于天河1A系统，天河1A是我国自主组建的大型GPU异构集群系统，该集群共有8100个节点，每一个节点接有2个6核Intel处理器和1个448核Tesla Fermi GPU。该集群采用定制的网络并使用胖树的交换结构，每一个节点具有80Gbps的理论带宽；将上述用于GPU异构集群的高性能FFT方法应用于天河1A系统，

包括如下步骤：

[0035] 首先使用数组维度表示方法，将初始数据记为数组数据$\frac{x}{[8192]}\frac{y}{[8192]}\frac{z}{[8192]}$；

[0036] 再将数组数据按照x维度分割为4096份，分布到天河1A系统的4096个节点上，这样，每一个节点的数据为2×8192×8192个单精度复数，此时数据记为

[0037] $\frac{\hat{x}_1}{[4096]}\frac{x_0}{[2]}\frac{y}{[8192]}\frac{z}{[8192]}$；

[0038] 天河1A系统的每个GPU进行x_0维度上2个2维的FFT，将y维度方向和z维度方向的数据进行计算；

[0039] 将数据进行转置，将y维度的数据做进一步的划分，得到：

[0040] $\frac{\hat{x}_1}{[4096]}\frac{x_0}{[2]}\frac{y_1}{[4096]}\frac{y_0}{[2]}\frac{z}{[8192]}$，

[0041] 在天河1A系统集群上直接使用copy子程序进行通信，将数据分布变换为

[0042] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\left[\frac{x_1}{[4096]}\frac{x_0}{[2]}\right]\frac{z}{[8192]}$，即

[0043] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\frac{x}{[8192]}\frac{z}{[8192]}$，

[0044] 这样，y维度和x维度得到交换，数据按照y_1维度分布在集群节点上，x维度方向的数据存储在各个节点之内；

[0045] 此时，各个节点可以分2(y_0维度)次，将8192×8192的数据拷贝到GPU设备存储器上，并进行正方形的数据转置，使得x维度的数据完全连续，得到：

[0046] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\frac{z}{[8192]}\frac{x}{[8192]}$；

[0047] 进行x维度的FFT计算；

[0048] 在使用CUFFT进行x维度的1维FFT计算之后，反方向重复整个数据移动过程，将数据恢复到初始的位置。

[0049] 与现有技术相比，本发明的有益效果是：

[0050] 已有高性能FFT研究集中于单机或传统集群，缺乏GPU异构集群的高性能FFT的研究。对于GPU异构集群，由于其具有更加复杂的多层存储器结构，传统集群上的FFT算法已经不再适合其体系结构的特征，难以取得更高的性能。

[0051] 与现有技术相比，本发明以天河1A系统为对象，针对GPU异构集群的多层存储器体系结构特征，提出适应性的数据维度划分和传输转置方法，通过数组维度类型程序设计，解决异构集群科学计算问题中复杂的数组维度处理问题：使用Parray语言的新型数组类型，可以在编码阶段对数组维度处理过程进行简洁的表示；同时，本发明还解决了如何使用数组维度类型程序设计方法和Parray语言实现天河1A系统上的大规模3维FFT，实现天河1A系统上的大规模3维FFT，该方法得到了良好的性能和可延展性。

附图说明

[0052] 图1是本发明提供的用于GPU异构集群的高性能FFT方法的流程框图。

[0053] 图2是传统集群的三维FFT方法数据处理示意图；

[0054] 其中，(a)为需要进行FFT计算的3维立方体数据；(b)为将3维立方体数据分布到4个处理器上后对y和z两个维度进行FFT计算的数据状态；(c)为将数据在4个处理器之间进行通信，使得x维度的数据连续以进行x维度FFT计算的数据状态。

[0055] 图3是使用本发明提供的GPU集群FFT方法和Intel公司数学库提供方法进行性能比较的结果示意图；

[0056] 其中，横坐标为进行性能测试的不同的数据维度大小规模；纵坐标为本发明的GPU集群FFT算法代码和Intel公司数学库提供方法的性能比较，单位为Gflops，即1G次浮点运算每秒。

[0057] 图4是使用本发明提供的GPU集群FFT方法处理的代码同Intel公司数学库提供方法进行性能加速比(即在增加参与计算节点的情况下，性能的提升情况)比较的结果示意图；

[0058] 其中，横坐标为使用天河1A的节点数目；纵坐标为其取得的性能，其单位为Gflops，即1G次浮点运算每秒。

具体实施方式

[0059] 下面结合附图，通过实施例进一步描述本发明，但不以任何方式限制本发明的范围。

[0060] 本发明提供一种GPU异构集群上的高性能FFT方法(PKUFFT)，将该方法应用于我国自主组建的大型GPU异构集群系统天河1A上，天河1A集群共有8100个节点，每一个节点接有2个6核Intel处理器和1个448核Tesla Fermi GPU，该集群采用定制的网络并使用胖树的交换结构，每一个节点具有80Gbps的理论带宽。

[0061] 本发明使用Parray语言实现GPU异构集群的高性能FFT方法。使用Parray语言表示维度变换所针对的数组包括自然数组、人工数组和混合数组；并使用Parray语言描述维度变换和转置过程的数据传输。表示方法分别如下：

[0062] a，自然数组

[0063] Parray语言中，自然数组指数组数据元素的逻辑下标和物理偏移相一致的数组类型。对于本发明中$\frac{x}{[a]}\frac{y}{[b]}\frac{z}{[c]}$这样的自然数组，Parray声明一个维度为a*b*c的数组类型，该数组为位于分页存储器中的浮点数数组，示例代码如下：

[0064]

```
1.$parray{paged float[a][b][c]}A
```

[0065] b，人工数组

[0066] 在Parray中，数组的物理存储与逻辑视图在概念上进行分离:类型提供且仅仅提供对数据的逻辑视图，与数据的物理存储空间无关；数组维度变换和转置通过人工数组实现。人工数组是指含有一个或多个人工维度的数组类型，人工维度来自于对已有数组类型维度的引用。对于一个正方形的数组类型A，其维度为b*c，其在内存中的转置表示为类型B的形式，其两个维度来自于类型A，但是维度的顺序进行了对调，人工数组的示例见下列代码：

[0067]

```
1. $parray {paged float[b][c]} A
2. $parray {[#A_1][#A_0]} B
```

[0068] c,线程数组和混合数组

[0069] Parray中,使用parray命令同样可以声明线程数组。

[0070] 如下代码声明一个集群上多个节点上进程构成的数组类型MYMPITA,其维度为a。

[0071]

```
1.$parray{mpi[a]}MYMPITA
```

[0072] Parray中,线程数组的维度可以和数据数组的维度结合在一起,构成一种特殊的人工数组,称为混合数组。混合数组可以用来表示分布的数据。下面混合数组表示的示例代码声明了一个混合数组类型T:

[0073]

```
1.$parray{[[#MYMPITA][#A_0]][#A_1]}T
```

[0074] 上述混合数组类型T中,它的T_0_0维度来自MPI进程数组类型MYMPITA,而T_0_1和T_1维度来自于分页内存数组类型DT。

[0075] 采用Parray语言表示数据的维度变换之后,将维度变换的数据传输到集群节点,具体地:

[0076] Parray中,可以通过调用copy子程序完成集群节点上甚至节点间的数据传输和转置,其语法为

[0077] $copy S(s)to T(t)

[0078] 其中,参数s和S为传输起始的数据地址和数组类型,参数t和T为传输目标的数据地址和数组类型,起始类型S和目标类型T需要具有相同的维度树结构。copy所完成的数据传输将根据起始类型S和目标类型T指示出的维度关系,将数据由起始地址s复制到目标地址t。

[0079] 在天河1A上,使用Parray书写的3维FFT代码最大运行到14336×14336×14336单精度复数的规模,共使用7168个节点,这是目前世界上最大规模的3维FFT运算。本实施实例以使用4096个节点计算8192×8192×8192规模的数据为例,描述Parray 3维FFT方法的步骤与应用:

[0080] 首先使用数组维度表示方法,初始数据记为$\frac{x}{[8192]}\frac{y}{[8192]}\frac{z}{[8192]}$,

[0081] 再将数组数据按照x维度分割为4096份,分布到4096个节点上,这样,每一个节点的数据为2×8192×8192个单精度复数。此时数据记为

[0082] $$\frac{\hat{x}_1}{[4096]}\frac{x_0}{[2]}\frac{y}{[8192]}\frac{z}{[8192]}$$

[0083] 然后,每个GPU先进行2(x0维度)个2维的FFT,将y维度方向和z维度方向的数据进行计算。由于计算的时间与数据传输的时间相比基本上可以忽略,在此直接使用CUFFT调用即可,无需使用已实现的更加优化的GPU FFT代码。

[0084] 然后,在进行x维度的FFT计算之前,需要进行转置,将y维度的数据做进一步的划分,得到

[0085] $\frac{\hat{x}_1}{[4096]}\frac{x_0}{[2]}\frac{y_1}{[4096]}\frac{y_0}{[2]}\frac{z}{[8192]}$,

[0086] 在集群上直接使用copy子程序进行通信,将数据分布变换为

[0087] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\left[\frac{x_1}{[4096]}\frac{x_0}{[2]}\right]\frac{z}{[8192]}$,即

[0088] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\frac{x}{[8192]}\frac{z}{[8192]}$,

[0089] 这样,y维度和x维度得到交换,数据按照y1维度分布在集群节点上,x维度方向的数据存储在各个节点之内。

[0090] 此时,各个节点可以分2(y0维度)次,将8192×8192的数据拷贝到GPU设备存储器上,并进行正方形的数据转置,使得x维度的数据完全连续,得到

[0091] $\frac{\hat{y}_1}{[4096]}\frac{y_0}{[2]}\frac{z}{[8192]}\frac{x}{[8192]}$,

[0092] 在使用CUFFT进行x维度的1维FFT计算之后,反方向重复整个数据移动过程,将数据恢复到初始的位置。

[0093] 以上方法采用Parray代码实现,代码如下所示:

[0094] Parray书写的集群3维FFT代码:

[0095]

```
1.  $parray {pinned float2[4096][[2][[4096][2][8192]]]} HSTS
2.  $parray {pinned float2[#HSTS_1]} HST
3.
4.  $subprog FFT3D_FORWARD(N, P, HST, DEV) {
5.      $parray {mpi[4096]} MYTA
6.      $parray {[[#MYTA][#HST_0][#HST_1_0][#HST_1_1]] [#HST_1_2]} S
7.      $parray {[[#HST_1_0][#HST_1_1][#MYTA][#HST_0]] [#HST_1_2]} T
8.      for(int slicei=0; slicei<2; slicei++) {
9.          $copy HST_1(host+$HST_0[slicei]) to DEV(dev)
10.         GPU_SAFE(cufftExecC2C(planb, dev, dev, CUFFT_FORWARD),"FFT X-Y")
11.         $copy DEV(dev) to HST_1(buf+$HST_0[slicei])
12.     }
13.     $copy S(buf) to T(host)
14.     for(int slicei=0; slicei<2; slicei++) {
```

[0096]

```
15.             $copy HST_1(host+$HST_0[slicei]) to DEVT(dev)
16.             $TRANSPOSE_OUTP(dev, dev2, $DEV)
17.             GPU_SAFE(cufftExecC2C(planc, dev2, dev2, CUFFT_FORWARD),"FFT Z")
18.             $TRANSPOSE_OUTP(dev2, dev, $DEV)
19.             $copy DEV(dev) to HST_1(buf+$HST_0[slicei])
20.         }
21.         $copy S(buf) to T(host)
22. }
23. $end
```

[0097] 其中，第1行声明自然数组类型HSTS，代表整个数据在集群上的维度划分；第2行声明自然数组类型HST，引用到HSTS_1维度，代表数据在一个节点上的维度划分；第5～7行，声明MPI进程数组类型MYTA以及混合数组类型S和T，注意S和T中的对应维度进行了调整，之后将使用其进行集群上数据的转置通信；进入FFT3D_FORWARD的执行代码，第8～12行，各个节点上分2次将y维度和z维度的8192×8192的数据传输到GPU上进行计算；第13行copy子程序利用之前声明的混合数组类型S和T进行算法中的集群之间的通信；第14～17行，各个节点将本地x维度的数据传输到GPU上进行1维FFT的计算；最后，反方向重复整个数据移动过程，将数据恢复到初始的位置。

[0098] 使用Parray实现的GPU集群3维FFT(PKUFFT)在天河1A上进行了性能测试，并与IntelMKL 10.3.1.048进行了比较。图3是使用本发明提供的GPU集群FFT方法和Intel公司数学库提供方法进行性能比较的结果示意图；其中，横坐标为进行性能测试的不同的数据维度大小规模；纵坐标为本发明的GPU集群FFT算法代码和Intel公司数学库提供方法的性能比较，单位为Gflops，即1G次浮点运算每秒。图3给出了它们在对不同规模的单精度复数数据做3维FFT时的性能；可以看出，PKUFFT的性能远远超出MKL。图4是使用本发明提供的GPU集群FFT方法处理的代码同Intel公司数学库提供方法进行性能加速比(即在增加参与计算节点的情况下，性能的提升情况)比较的结果示意图；其中，横坐标为使用天河1A的节点数目；纵坐标为其取得的性能，其单位为Gflops，即1G次浮点运算每秒。图4显示出，与MKL相比较，PKUFFT具有更好的性能延展性。

[0099] 需要注意的是，公布实施例的目的在于帮助进一步理解本发明，但是本领域的技术人员可以理解：在不脱离本发明及所附权利要求的精神和范围内，各种替换和修改都是可能的。因此，本发明不应局限于实施例所公开的内容，本发明要求保护的范围以权利要求书界定的范围为准。

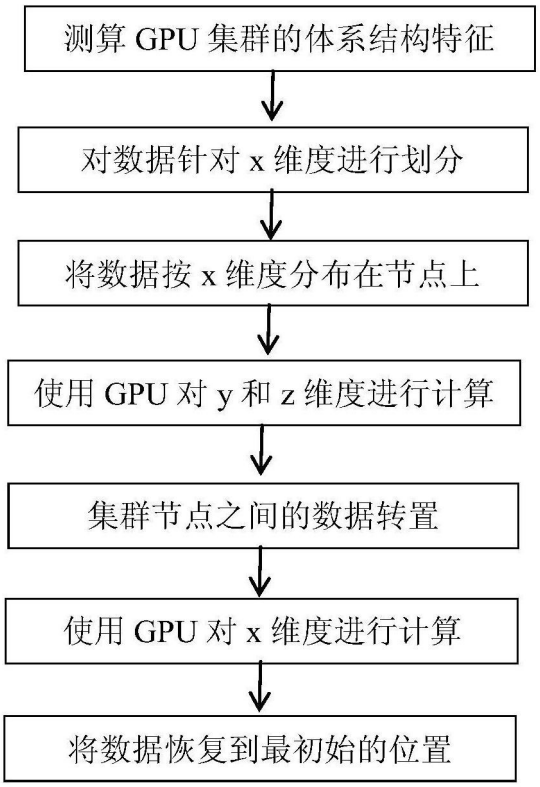

测算 GPU 集群的体系结构特征
对数据针对 x 维度进行划分
将数据按 x 维度分布在节点上
使用 GPU 对 y 和 z 维度进行计算
集群节点之间的数据转置
使用 GPU 对 x 维度进行计算
将数据恢复到最初始的位置

图1

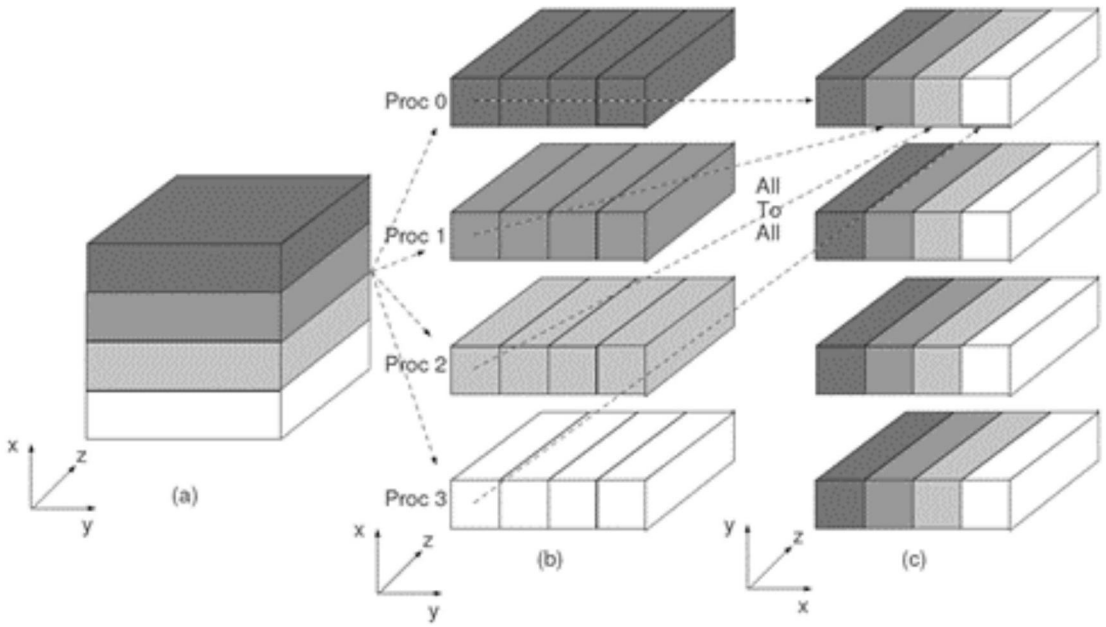

Proc 0
Proc 1
Proc 2
Proc 3
All
To
All
x
z
y
(a)
(b)
(c)

图2

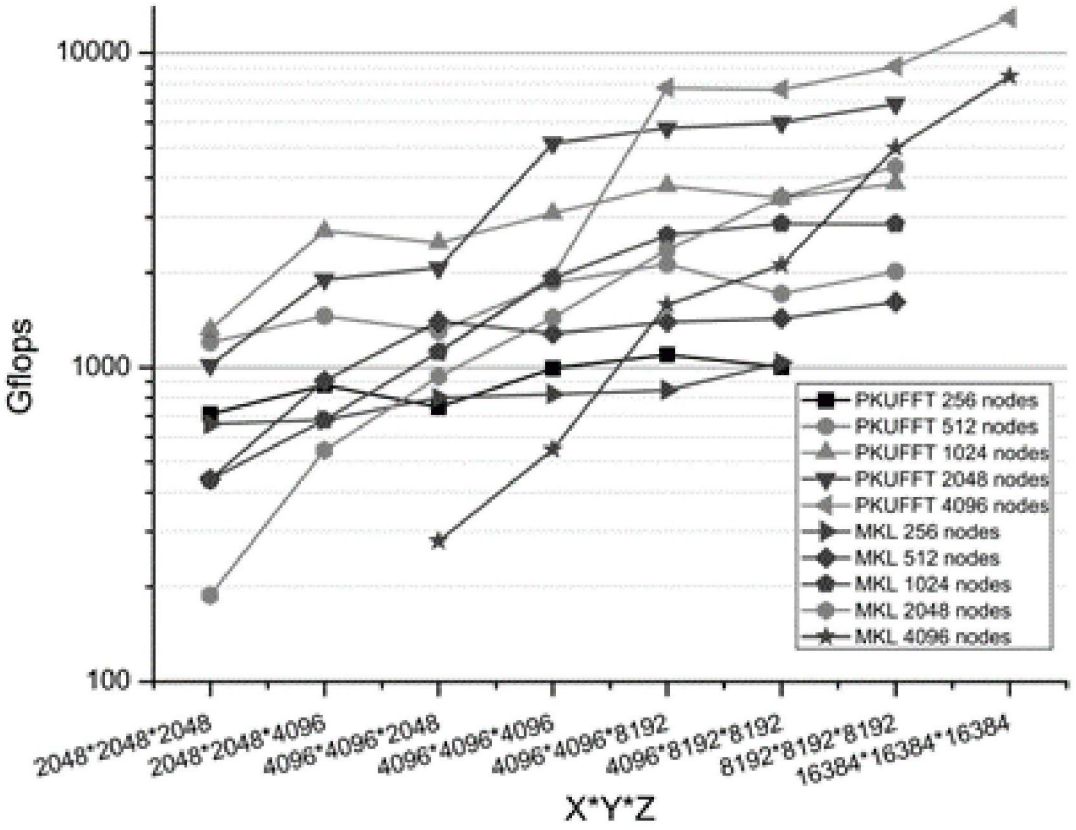

10000
1000
100
Gflops
PKUFFT 256 nodes
PKUFFT 512 nodes
PKUFFT 1024 nodes
PKUFFT 2048 nodes
PKUFFT 4096 nodes
MKL 256 nodes
MKL 512 nodes
MKL 1024 nodes
MKL 2048 nodes
MKL 4096 nodes
2048*2048*2048
2048*2048*4096
4096*4096*2048
4096*4096*4096
4096*4096*8192
4096*8192*8192
8192*8192*8192
16384*16384*16384
X*Y*Z

图3

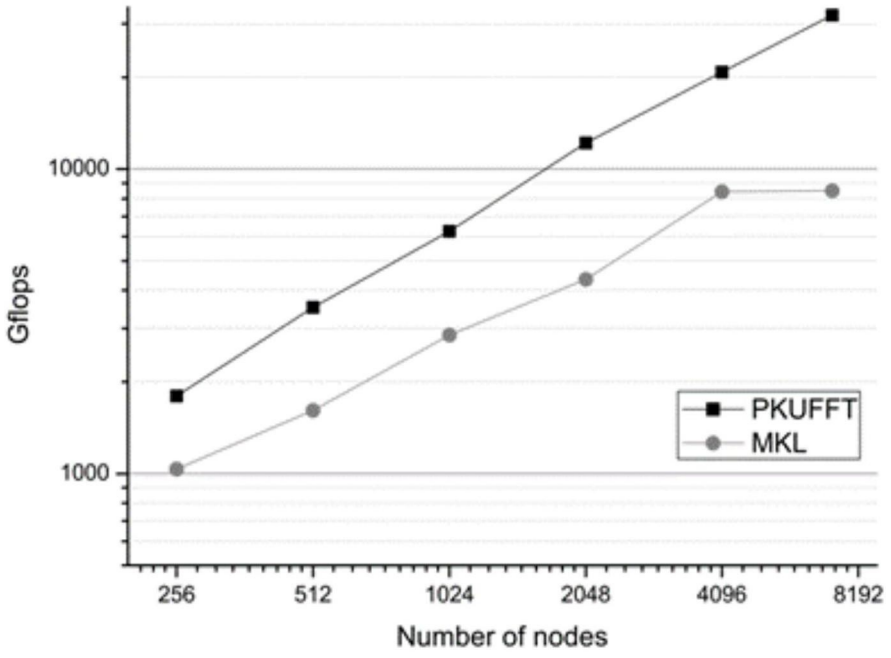

10000
1000
Gflops
PKUFFT
MKL
256
512
1024
2048
4096
8192
Number of nodes

图4