第 31 卷 第 4 期
2013 年 11 月

天 文 学 进 展
PROGRESS IN ASTRONOMY

Vol. 31, No. 4
Nov., 2013

doi: 10.3969/j.issn.1000-8349.2013.04.05

GPU 并行计算技术在赫歇尔天文台远红外巡天数据处理中的应用

朱佳丽[1,2]，黄茂海[1]

(1. 中国科学院 国家天文台，北京 100012； 2. 中国科学院大学，北京 100049)

摘要：赫歇尔远红外银道面巡天数据由于其高分辨率和大尺度特性而具有海量性，使用图像处理单元 (GPU) 提高数据挖掘处理效率是海量数据处理中的一个重要环节。以 GPU 在实际研究工作中的使用方法和思路为着眼点，结合赫歇尔远红外数据处理过程中 GPU 的具体应用，介绍了 GPU 并行数据处理技术在星际介质物理性质和三维结构研究中的应用。对赫歇尔远红外数据的处理挖掘使用 GPU 加速的有两个部分：银道面温度图的每像素拟合和计算星际介质三维模型中单元格对星光的吸收。在这两个应用中，分别使用了 CUDA 和 PyOpenCL 调用 GPU 进行并行计算。这种使用 GPU 提高数据处理效率的思路与方法对与两个实例相类似的数据处理问题具有借鉴意义。

关 键 词：GPU；并行计算；赫歇尔银道面红外巡天；数据处理方法

中图分类号：N37，P141.91 **文献标识码**：A

1 引 言

由于天文学数据尤其是高分辨率的巡天观测数据的海量性，并行计算技术成为重要的提高天文数据处理分析速度的重要措施。图像处理单元 (graphics processing unit，简称 GPU) 的出现使得大量的浮点数运算得以在较短的时间内完成。GPU 的概念最初由英伟达公司 (NVIDIA) 提出，从此图形显示卡超越了仅显示图像的作用，成为一个独立于 CPU 的计算单元。目前，GPU 的生产商有 NVIDIA 和 AMD。由于 CPU 的核心架构不利于其向多核心的发展，而 GPU 的众核架构使 GPU 的计算核心要远多于 CPU 的运算核心。以 NVIDIA Tesla C2075 为例，其核心数达 448 个，单精度运算速度达 1030 Gflops，运算能力远超 CPU。GPU 天生的多线程并行性和相对廉价性，在并行运算方面有独到的作用。

赫歇尔银道面红外巡天项目 Hi-GAL[1](Herschel infrared Galactic Plane Survey) 的 SDP

收稿日期：2013-05-20； **修回日期**：2013-06-26

资助项目：中国科学院知识创新工程重要方向项目 (KJCX2-YW-T20)

(science demonstration phase) 数据包括两幅 2° × 2° 的图，图的中心坐标为 $l = 30°$，$b = 0°$ 和 $l = 59°$，$b = 0°$。每一幅图都有 5 个波段的观测数据，分别为 PACS 70 μm 和 160 μm，SPIRE 250 μm，350 μm 和 500 μm，单波段图像的像素数目达 $10^5 \sim 10^6$ 量级。Hi-GAL 数据的海量性对提高数据的处理速度提出了新要求。在赫歇尔远红外数据的处理挖掘过程中有两个部分的数据处理流程具备较高并行度，可以使用 GPU 对数据进行并行处理。

第 2 章对使用 GPU 进行并行计算的技术进行简单介绍，在第 3 章中分别介绍 GPU 在赫歇尔远红外数据处理中一个并行运算的简单例子和一个较复杂的例子。本文着重于介绍 GPU 并行数据处理技术在星际介质物理性质和三维结构研究中的应用，GPU 的硬件架构和工作模式等则不在本文主要讨论范围之内。

2 利用 GPU 进行并行计算的技术

要使用 GPU 进行科学计算，必须使用针对 GPU 的编程语言。NVIDIA 专门开发了 CUDA (Compute Unified Device Architecture) 用于 GPU 的运算。CUDA 的语法基于 C 和 C++ 语言，所以只要有 C 语言基础就很容易入门，而不必细究 GPU 的内部细节。具体的编程方法可参考 CUDA 的编程指南 [2]。由于 CUDA 仅适用于 NVIDIA 的 GPU，其跨硬件的通用性较差。另一种重要的 GPU 编程环境是 OpenCL[3] (Open Computing Language)，最初由苹果公司开发，之后 NVIDIA、AMD、IBM 和 Intel 等厂商也加入了 OpenCL 规范。OpenCL 是第一个面向异构系统，具有通用目的，可以并行编程的开放式免费标准，也是一个统一的编程环境。OpenCL 源程序可以在不同厂商生产的 GPU 和 CPU 上编译运行，大大提高了代码的可移植性。在具体的编程语言上，和 CUDA 一样，OpenCL 的语法也是基于 C 语言。

不论用 CUDA 还是 OpenCL 编程，源程序都包含两部分 [2, 3]：(1) 在主机即 CPU 上执行的代码，称为主程序 (host code)；(2) 在 GPU 设备上执行的代码，称为内核程序 (kernel)。在 CUDA 中，当 CPU 调用 GPU 时，内核程序将在多个线程 (thread) 中同时执行，一定数目的线程组成一个线程块 (block)；OpenCL 中与线程和线程块对应的概念是工作节点 (work item) 和工作组 (work group)。在内核程序中，必须要先获得线程和线程所在块的索引 (ID) 才能开始让 GPU 执行运算。当使用 GPU 进行数据并行运算时，数据与数据之间地位基本平等，所以并行算法中要求处理每一个数据的线程行为基本一致。当并行算法中选择分支较多时，由于不同的线程执行的选择分支不同，会大大降低 GPU 的运算效率。

3 应用 GPU 并行处理 Hi-GAL 数据

在求解银道面温度图和构建星际介质三维模型时，由于需要处理的数据量极大和部分数据处理流程具有较高的可并行性，我们使用了 GPU 进行并行运算以加速计算过程。基于研究当时的硬件和软件条件，在求解温度图时使用的是 CUDA C 语言，在三维模型计算时则使

用了 PyOpenCL 语言 (OpenCL 的 Python 版本)。其中，求解温度图是简单经典的并行处理例子，而星际介质三维模型中 GPU 的应用是更为复杂的并行处理的例子。本文目的在于结合具体应用介绍 GPU 在赫歇尔远红外海量数据处理中的应用，因此在下文中将对科学内容大大简化，去除了所有不必要的物理细节。

3.1　GPU 并行运算技术在银道面温度图拟合中的应用

3.1.1　计算需求和面临的困难

尘埃颗粒吸收远紫外波段辐射后发射红外辐射，处于热平衡状态的大尺寸尘埃颗粒 ($\gtrsim 0.01$ μm) 发射波长 $\gtrsim 50$ μm 的远红外辐射 [4]。通过拟合尘埃的远红外辐射可以得到尘埃的热平衡温度。对 Hi-GAL 图像的每一个像素的远红外辐射进行拟合则可得到银道面温度图。

在使用 CPU 进行拟合时，需要对所有像素依次进行拟合，耗时和需要处理的像素数目成正比。对于高分辨率的赫歇尔远红外巡天数据而言，由于像素数目众多，完成一幅温度图拟合耗时较大。以 Hi-GAL $l = 30°$ 区域为例，使用 4 个波段数据，包括 PACS 160 μm，SPIRE 250 μm，350 μm 和 500 μm，将所有的图像都被平滑到和 SPIRE 500 μm 相同的分辨率后，该区域的像素总数高达 313 917 个。在数据处理过程中，像素和像素之间相互独立，并且每一个像素的拟合过程相同，因此温度图的求解具备极高的可并行性。如果使用 GPU 对所有像素进行并行处理则可大大提高拟合效率。

3.1.2　GPU 并行计算程序的设计和实现

温度图拟合所使用的模拟数据构成一个二维网格，网格大小为尘埃温度个数 × 比例因子个数。我们使用气体云团数值模型 CLOUDY[5]，计算了一系列热平衡温度下辐射强度归一化的尘埃远红外辐射，其中不同的温度对应不同的能谱分布形状，比例因子则用于和这个辐射强度归一化的能谱分布相乘，从而得到远红外波段辐射的绝对值作为模拟数据。拟合方法采用网格遍历，即每一个像素的红外辐射观测值都要和网格上所有的点对应的模拟数据相比较，以得到尘埃温度的最优拟合值。由上述可知，每一个像素的拟合过程相同，因此可以用 GPU 的单个线程来处理单个像素的拟合，所有的线程执行相同的内核程序。

整个拟合程序使用 CUDA C 写成，根据程序运行中涉及的硬件设备，可将源代码分为 3 大部分：(1) 数据的读入和预处理，仅涉及 CPU；(2) CPU 将必要的数据拷贝到显存中并调用 GPU 进行运算，CPU 将 GPU 处理完的数据从显存拷贝到本地并由 CPU 做后续处理；(3) 在 GPU 上进行的并行运算。

下面是部分源程序，旨在叙述调用 GPU 进行并行运算的逻辑和方法，源程序有大量删节，仅列出了和使用 CPU 处理数据的不同之处，对应的数据处理流程图参见图 1。

```
/* 在 GPU 上运行的内核程序 */
__global__static void pixels(float *gpudata,float *res)
{
        long which;
        const int tid=threadIdx.x;/* 获取线程 ID*/
        const int bid=blockIdx.x;/* 获取块 ID*/
```

```
        /* 在线程上运算，一个线程对应一个像素 */
        for(which=bid*THREADS_NUM+tid;which<LINE; which+=THREADS_NUM*BLOCK_NUM){
            /* 数据并行处理部分，和 CPU 对每像素的处理过程相同 */
            …
         }
}
/*CPU 运行部分 */
main()
{
        /* 读取数据 */
        …
        /* 对数据做预处理以备 GPU 使用 */
        …
        /* 将数据拷贝到 GPU 显存 */
        cudaMalloc((void**) &gpudata, sizeof(float)*WIDTH*LINE);
        cudaMemcpy2D(gpudata, sizeof(float)*WIDTH, sed30, sizeof(float)*WIDTH,sizeof(float)*WIDTH, LINE,
cudaMemcpyHostToDevice);
        …
        /* 调用 GPU，<<<...>>> 中指出要使用的线程块数目和每个线程块中的线程数，以及其他必要的
内核运行参数。括号内列出传递给 GPU 的数据 */
        pixels<<<BLOCK_NUM,THREADS_NUM,parameters>>>(gpudata, res);
        /* 从 GPU 显存中将处理后的数据拷回到 CPU 内存 */
        cudaMemcpy2D(result, sizeof(float)*WIDTH, res, sizeof(float)*WIDTH, sizeof(float)*WIDTH, LINE,
cudaMemcpyDeviceToHost);
        …
}
```

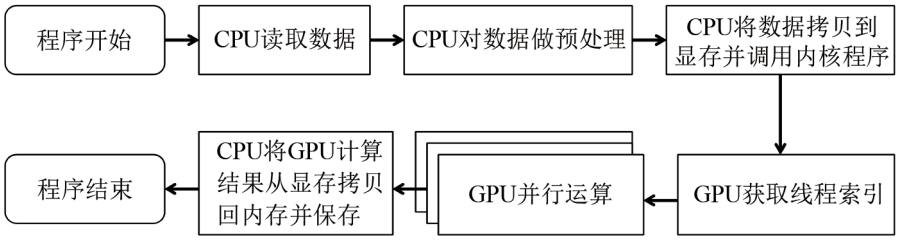

图 1 数据处理流程

3.1.3　GPU 加速的显著效果

在研究初期，模拟数据的网格大小为 87 × 500 ，硬件条件是笔记本自带的 NVIDIA GeForce 9400M。在 IDL7.1 开发环境下，使用 CPU (Xeon E5620，使用 1 个核) 完成整张图拟合耗时约 34 h，GPU (NVIDIA GeForce 9400M) 完成整张图拟合耗时 0.5 h，加速比为 68 (见表 1)。NVIDIA GeForce 9400M GPU 只能进行单精度运算，由于网格较粗，即数据点间隔较大，GPU 拟合结果和 CPU 拟合结果一致。在研究中后期，随着网格精细度和模拟数据范围不断加大，硬件也相应升级。在 IDL7.1 开发环境下，使用 CPU (Xeon E5620，使用 1 个核)，在网格第一个参数固定的情况下，完成 1000 个像素的拟合耗时约 45 s，完成 l = 30° 区域一幅图的拟合耗时约 827 h。在 CUDA 开发环境下，使用高性能的 NVIDIA Tesla C2075 GPU 对所有像素进行双精度并行运算，9 min 就可以完成一个 4 平方度区域的拟合，加速比为 5513 (见表 2)。

表 1　CPU 和较低性能 GPU 的运算性能比较

运算平台	耗时 (t/min)	加速比
CPU(Xeon E5620，使用 1 个核)	2040	68
GPU(NVIDIA GeForce 9400M)	30	

表 2　CPU 和较高性能 GPU 的运算性能比较

运算平台	耗时 (t/min)	加速比
CPU(Xeon E5620，使用 1 个核)	49620	5513
GPU(NVIDIA Tesla C2075)	9	

从表 1 和表 2 可见 GPU 在处理高分辨率的大尺度图像上占有绝对的时间优势。即使在硬件配置较低的条件下，使用 GPU 仍能极大地提升数据处理效率。图 2 呈现了使用 NVIDIA Tesla C2075 GPU 拟合得到的 l = 30° 区域的温度图 [6]。

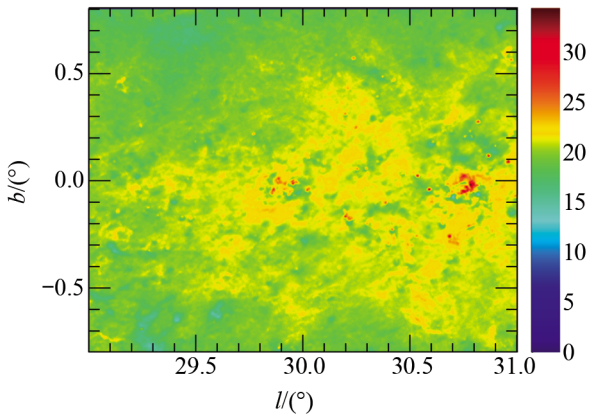

图 2　l = 30° 区域温度

3.2 GPU 并行运算技术在星际介质三维模型中的应用

3.2.1 科学问题概述和面临的困难

为了进一步对 Hi-GAL 数据进行挖掘，我们构建了星际介质的三维模型，将该模型计算得到的三维空间星际介质的物理性质积分到二维银道面，并和观测数据相比较。模型的几何结构由三维网格构成，整个三维网格由众多的单元格组成。以三维直角坐标系为例，当网格的分辨率为 10 pc×10 pc×10 pc 时，每一个单元格都是以 10 pc 为边长的立方体。

针对银道面数据，实际计算中使用球坐标 (spherical coordinates)，坐标的 (r, θ, ϕ) 对应了 $(d, \pi/2 - b, l)$，其中 d 是离太阳距离，l 和 b 分别是银经和银纬。结合 GRS 分子云巡天数据 [7] 以及目标区域中氢二区的物理化学性质，当采用球坐标的三维网格分辨率为 100 pc × 0.2° × 0.2° 时，l = 30° 区域中充满分子气体的单元格数目有 4493 个。三维模型计算每一个充满分子气体的单元格中气体和与气体成协的尘埃对三维网格中氢二区的远紫外辐射的吸收。这种三维空间的吸收被投影到银道面形成一幅模拟的二维吸收图像，该图随星际介质的三维密度分布的改变而改变。将模拟的二维吸收图像和实际的二维吸收图像 (通过温度图拟合获得) 比较，寻找星际介质三维密度分布的最优拟合值。在拟合过程中使用马尔科夫链蒙特卡罗方法来对单元格气体密度值抽样，并通过足够多次的迭代使马尔科夫链达到收敛以得到最优拟合值。

在三维模型的计算中 GPU 并行运算技术的应用是不可或缺的环节。CPU 需要依次计算所有单元格对辐射的吸收，总耗时和单元格的数目成正比，当三维网格分辨率为 100 pc × 0.2° × 0.2° 时，CPU (Xeon E5620，使用 1 个核) 在 Python 2.7 编程环境下计算一次模拟二维吸收图即单次迭代耗时约 6.8 min。当网格的角分辨率和 Hi-GAL 数据 500 μm 的分辨率 (约 0.01°) 相同时，单元格数目更是高达百万量级，使用 CPU 计算单次迭代就需要几十个小时。由于自由参数数目 (相当于单元格的总数) 众多，马尔科夫链达到收敛需要百万次甚至千万次的迭代，仅使用 CPU 参与计算将无法在合理的时间内完成拟合。由于计算每一个单元格对辐射吸收的运算流程高度相似，并且每单元格的计算相互独立，所以计算三维网格中单元格对辐射的吸收过程具有较高可并行性。如果使用 GPU 并行计算每一个单元格对远紫外辐射的吸收，将大大提高三维模型的计算效率，从而在合理的时间内完成拟合。

3.2.2 程序设计

在每一次迭代中，GPU 负责并行计算每一个单元格对远紫外辐射的吸收，CPU 负责对自由参数抽样和计算拟合优度。在完成一定数量的迭代后，CPU 需要判断马尔科夫链是否收敛以确定是否继续迭代。在本例中，整个拟合过程需要 CPU 和 GPU 的共同参与，而温度图拟合中，拟合过程全部在 GPU 上完成。三维模型的程序运行流程详见图 3。从耗时上看，使用 NVIDIA Tesla C2075 GPU，单精度运算一次迭代仅需 0.043 s (20 000 次迭代的单次平均耗时)，并且在 GPU 上的耗时占单次迭代全部时间的 95% 以上，和仅使用 CPU 运算比较加速比达 9535 (见表 3)。由于 GPU 单精度运算速度高于双精度运算速度，在确保计算结果准确度的前提下，可采用单精度运算来提高运算效率。比较 GPU 单精度运算结果和双精度运算结果，两者差别小于 10^{-5}，远低于观测数据误差，可忽略不计。

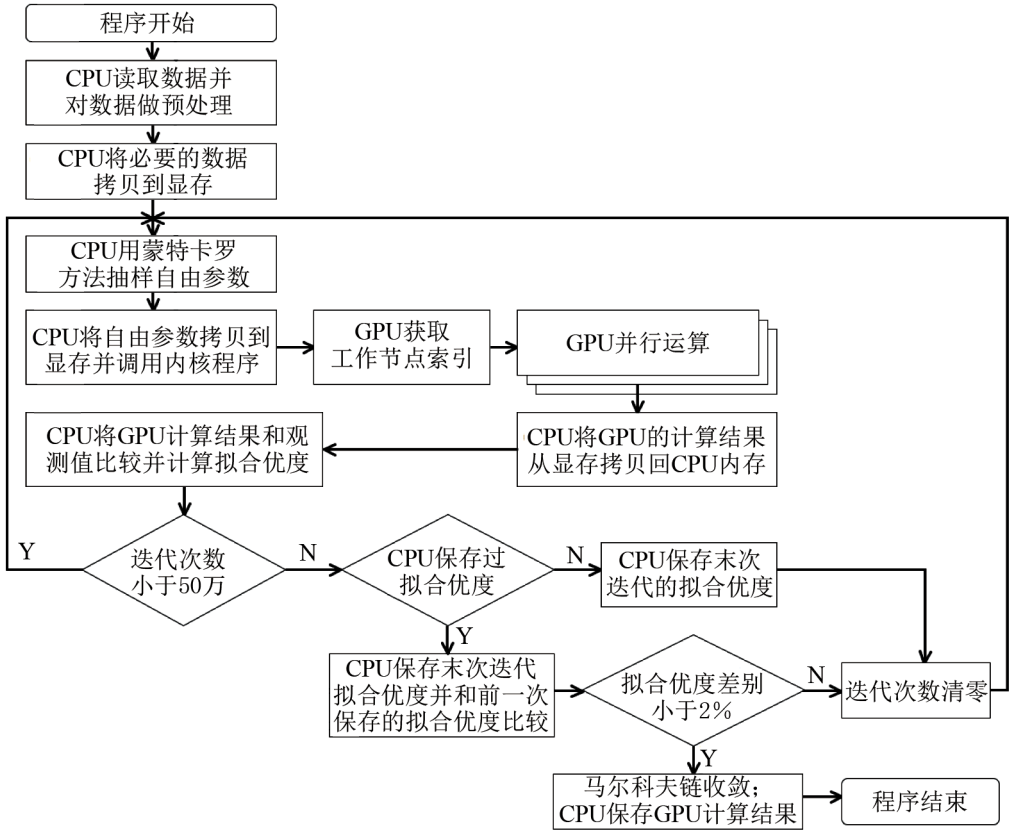

图 3　数据处理流程

表 3　CPU 和 GPU 运算性能比较

运算平台	单次运算平均耗时 (t/s)	加速比
CPU(Xeon E5620，使用 1 个核)	410	9535
GPU	0.043	

在三维模型的应用中，使用了 PyOpenCL 来调用 GPU 进行计算。PyOpenCL 是一个 Python 包，使用户能在 Python 中使用 OpenCL 的所有功能。除了具体的函数、语法不同外，使用 OpenCL 调用 GPU 和上述使用 CUDA C 调用的流程相同，具体编程方法可参考 PyOpenCL 的使用文档[8]。和上一个例子中使用线程和线程块的索引不同的是，在本例中使用了三维的工作节点索引，即每一个工作节点都对应一个三维的索引号 (i, j, k)，其中 $0\le i<\mathrm{MAX}(i)$，$0\le j<\mathrm{MAX}(j)$，$0\le k<\mathrm{MAX}(k)$。$\mathrm{MAX}(i)$、$\mathrm{MAX}(j)$ 和 $\mathrm{MAX}(k)$ 是工作节点在每一维度的最大索引值，由 GPU 的硬件性能参数决定。同时也将三维模型的所有单元格按其中心坐标进行编号，如中心坐标为 (500 pc, 0.4°, 0.6°) 的单元格的三维索引为 (5, 2, 3)。将单元格和工作节点通过三维索引号一一对应，使得一个工作节点处理一个单元格的运算，所有的工

作节点运行相同的内核程序。

3.2.3 CPU 对数据的预处理

在看到 GPU 强大的并行计算能力的同时，也不可忽视 CPU 对要传递给 GPU 的数据的预处理。在三维模型的构建过程中，利用了 CPU 对 GPU 运算中涉及的数据做预处理来提高 GPU 的运算效率。这种预处理包含两个方面：

(1) 对数据结构的优化。因为 GPU 中每个线程的行为要尽可能地保持一致，若各个线程对应的选择分支差别太大，则会大大降低并行运算的效率。在三维模型中，每个单元格相对独立，但是目标单元格对辐射的吸收受到单元格和辐射源之间的光线路径上所有单元格的影响；所以，线程在计算不同单元格的吸收时所遇到的分支是不尽相同的。要减少分支的不一致性，可以利用 CPU 对数据做预处理，使数据的结构和组成能够提高各个线程行为的一致性。

(2) 提取 GPU 运算中不随自由参数改变的运算部分，由 CPU 预处理该部分的运算，直接将计算结果传递给 GPU，而不必在迭代时由 GPU 重复计算。

以上的预处理不涉及迭代部分，因此在程序运行过程中仅执行一次。利用预处理提高 GPU 运算效率需要结合科学目标、模型数值算法等具体问题进行具体分析，优化的程度应具体问题而异。

3.3 未来发展

在 GPU 工作模式方面，结合三维模型数值计算的特点，将对并行程序做进一步优化。涉及的优化包括两个方面 [3]：(1) 数据传输的优化，即尽量避免 CPU 和 GPU 之间不必要的数据传输；(2) 内存访问的优化，同一个工作组的工作节点尽量使用组内的共享存储 (Local Memory)，减少访问存取速度相对较慢的全局存储器 (Global Memory) 的次数。

在 GPU 硬件性能方面，GPU 本身的运算能力也是限制计算效率的原因。在三维模型的计算中，我们主要使用了 NVIDIA Tesla C2075 GPU，并对运算能力更高的 K20m GPU 进行了测试。由于一次计算耗时不到 0.1 s，为减少统计误差，采用多次迭代来计算单次迭代的平均耗时，并总结在表 4 中。从表 4 中可见 K20m GPU 相对于 C2075 GPU 的单精度运算速度提高了约 40%。另外 GPU 工作节点的数目受到硬件性能的限制，C2075 GPU 可提供的最大工作节点数在三个维度上依次为 1024, 1024 和 64，所以当三维模型的网格分辨率进一步提高使得需要并行处理的单元格数目超过最大工作节点数时，GPU 将对数据分批先后进行处理。例如需要处理的单元格有 2000 个，GPU 能提供的工作节点为 1000 个，那么 GPU 并行计算完前 1000 个单元格后再处理后 1000 个单元格，在运算耗时上增加了一倍。所以为了不降低并行运算的效率，在较高分辨率情况下，我们将需要能提供更多工作节点的 GPU 或使用多个 GPU 进行运算。在使用多个 GPU 进行运算时，需要在数据预处理中使用 CPU 对目标区域分块，一块 GPU 处理一块子区域，子区域内的单元格数目不超过单块 GPU 的工作节点数目。在三维模型这个应用中，不同 GPU 之间不需要数据传输。当实际应用中采用多块 GPU 进行运算时，若不同 GPU 之间需要进行大量数据传输，就会降低数据处理效率，此时需要和采用单块 GPU 时的运算效率相比较，取效率相对较高者进行运算。

表 4　K20m 和 C2075 的运算速度比较：单次迭代平均耗时 (t/s)

迭代次数	K20m	C2075	K20m/C2075
20 000	0.027	0.043	63%
200 000	0.029	0.049	59%

4　总结和展望

本文以 GPU 在赫歇尔远红外海量数据处理中的两个具体应用作为例子，介绍了 GPU 并行数据处理技术在星际介质物理性质和三维结构研究中的应用。这种通过使用并行运算加快数据处理速度的方法和思路具有一定的通用性，对于和文中实例相似的问题均具有参考价值。在看到 GPU 强大并行能力的同时，也不可忽视 CPU 的作用。面对具体问题，应该将问题分解，提取出可并行部分让 GPU 计算，CPU 则负责运算其余部分并对要传递给 GPU 的数据做预处理以提高 GPU 运算效率。GPU 的运算能力受到其硬件性能的限制，应从具体问题出发，选择性能适当的 GPU 以提高运算效率。

致谢

对 Rainer Spurzem 教授、Peter Berczik 博士、Peter Schwekendiek 博士在 NVIDIA Tesla K20m GPU 的三维模型运算测试中提供的帮助深表感谢。

参考文献：

[1] Molinari S, et al. A& A, 2010, 518: L100
[2] NVIDIA Corporation. NVIDIA CUDA C Programming Guide version 4.2, http://wwwteor.mi.infn.it/~vicini/CUDA/doc/CUDA_C_Programming_Guide.pdf, 2012
[3] Advanced Micro Devices, Inc. AMD Accelerated Parallel Processing OpenCL Programming Guide, http://www.primeval-slayer.com/amd-accelerated-parallel-processing-opencl-programming-guide/read/41855/, 2012
[4] Draine B T. ARA&A, 2003, 41: 241
[5] Ferland G J, Korista K T, Verner D A, et al. PASP, 1998, 110: 761
[6] Zhu J L, Huang M. 2013, in preparation
[7] Roman-Duval J, Jackson J M, Heyer M, et al. ApJ, 2009, 699: 1153
[8] Andreas K. PyOpenCL documentation, http://documen.tician.de/pyopencl/, 2013

Applying GPU Parallel Computing Technologies to Process Herschel Far Infrared Galactic Plane Survey Data

ZHU Jia-li[1,2], HUANG Mao-hai[1]

(1. National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100012, China; 2. University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: The Hi-GAL (Herschel infrared Galactic Plane Survey) images provide data with extraordinary spatial coverage and resolution for studying the FIR emission in the Galactic Plane. Graphics processing unit (GPU) parallel computing technologies are well suitable for accelerating processing and mining of this massive data. We illustrate the application of GPU parallel computing technologies in two examples of Hi-GAL data processing. We spare the unnecessary physical details and focus on the method of using GPU in Herschel infrared data processing.

In the first example, we demonstrate a simple and straightforward application of GPU parallel computing technologies by fitting the far-infrared spectral energy distribution of the dust continuum emission in the Hi-GAL $l = 30^\circ$ field. There are over 3×10^5 pixels in image of the $l = 30^\circ$ field. The fitting procedure for every pixel is performed in parallel by a GPU. Comparing the time-cost for fitting the entire image, the acceleration factor of the build-in GPU on a low performance laptop is 68, and a specialized GPU is 5513 times faster than a Xeon E5620 with one core.

In the second example, we demonstrate a more sophisticated application of GPU parallel computing technologies. Based on the Hi-GAL $l = 30^\circ$ field data, the distribution of molecular clouds derived from GRS (Galactic Ring Survey) data, and the properties of H II regions, we construct a 3D model of the interstellar medium to calculate the absorption of dust grains associated with molecular clouds. The resolution of the 3D model is 100 pc $\times$ 0.2° $\times$ 0.2° for a spherical grid. For this resolution, there are 4493 cells in total responsible for absorbing FUV photons. The absorption of these cells is calculated in parallel by a GPU. The resulted absorption is then compared with observations using Monte Carlo fitting method. In every iteration, CPU samples the free parameters and computes the goodness of fitting. The GPU part of the calculation is 95% of the total time. Comparing the time-cost for one iteration, NVIDIA C2075 GPU is 9535 times as fast as a Xeon E5620.

Key words: GPU; parallel computing; Hi-GAL; data analysis