**并行程序设计大作业-求出曼德博集合**

**1. 学号、姓名**

SY1921120 李康佳

**2. 实现**

**遇到的问题与解决措施**

一：编写mpi\_static.c文件时，尝试了好几种思路。最初，采用发送接收方式编写程序，但是效果不理想，无法生成图形文件，调试发现，生成图形文件的代码根本没执行到，后来发现是通信产生死锁，导致有的代码执行不到。在编写计算块的像素代码时，形式参数没有设置好，导致程序运行不出理想的结果，后来更改形参，代码就可以正确运行了。在尝试每个进程各自运算一部分像素时，没有利用MPI的消息传递机制，想当然的以为各个进程计算的结果就是全局结果而不用统一收给某一个进程，导致了图片只有1/进程数的区域是有效的。最后成功运行是通过MPI的gather函数来完成的。

二：编写mpi\_dynamic时，想利用较少的通信达成目的，但是经过一番尝试，发现不成功，程序会出现各种错误或不理想的结果。最后重新整理了思路，按照主进程调度，副进程执行任务，相互通信，程序得以顺利运行。

三：编写hybrid.c时，一开始忘了include<omp.h>,后来发现还是没有加速，仔细检查代码，发现是因为动态的MPI调度，每次都是单行，没法再对行omp并行了，于是改为对列进行omp优化，效果很好。

**各种版本的优化是如何实现的**

mpi\_static,c优化方式是：所有进程计算若干行的像素值，最后由0号进程进行收集，绘制曼德博集合图片。

mpi\_dynamic优化方式是：0号进程负责调度，其余进程负责具体计算任务。首先0号进程给所有副进程发送一个行号，让每个副进程去执行任务，当有副进程执行完任务后，发回结果给0号进程，如果此时计算任务尚未分配完，则0号进程再给该进程发送一个递增的行号，让该进程去执行。循环如此执行，直到所有任务都完成。

omp.c的优化方式是：将计算每行像素值的任务，动态分配给各个线程执行。

hybrid.c优化方式是：mpi有关的部分，采用动态调度的方式；omp有关的部分，当计算固定行像素值时，每一个像素点的值，动态调度线程去执行。

**是如何将任务进行分块的**

静态MPI调度，采用将总行数分为进程数个部分，每个进程执行连续的几行像素值计算。动态MPI调度，将任务按照单行方式划分。Omp将任务按照单行方式划分。混合优化，采用先按单行划分，再按每行的单个像素点方式划分。

**是如何提升运行速度与增强程序扩展性的**

采用并行计算，让每个核的空闲时间变少来提升运行速度。关于扩展性，对于静态调度来说，如果像素点的行数或列数不是核数的整数倍，可以将最后剩余的零头单独处理。对于动态调度，则不存在这个问题。

如果是提升并行计算的可扩展性，可以通过增大计算任务规模实现。

**3. 分析与实验**

**是如何测量程序运行时间的**

在每个程序执行的开始和结束记录时间，然后输出差值，就可得出程序运行时间。

**请用 Excel 或者其他类似工具作出比较图，比较** 3 **个不同的因素所带来的的**加速比**变化：** 进程的数量，线程的数量和优化版本。 **写清楚每张图的纵轴横轴是什么，  
是在什么版本运行下的。**

**分析加速比随自变量变化的趋势，及发生这种变化的原因。**

静态mpi调度随着进程数的增加，加速比提升变得缓慢，原因应该是进程负载不均衡导致的；动态mpi由于主进程不执行计算任务，所以进程数为2时加速比还约为1，但当进程数继续增加时，加速比提升迅速。

**4. 结论**

**通过上述比较，得出最佳的优化方案是什么，该方案的优势是什么**

最佳的优化方式是hybrid，该方案的优势是既对行动态调度并行，又对列动态调度并行。

**在混合编程中，进程数量和线程数量该如何分配才能达到最佳效果**

对于该程序来说，我认为，让行数/进程数近似等于列数/线程数，效果最佳。