# 并行结算实验报告

**赵东杰 P18206023 手机号：13121595665 邮箱：[zdj8023first@163.com](mailto:zdj8023first@163.com)**

**开发前准备**

开发配置：4核8线程

开发环境：Ubuntu 18.04 LTS

开发平台：Clion

开发语言：c++

**1 稀疏矩阵求解**

通过配置项目的CMakeLists.txt来配置openmp环境

# 配置openmp  
FIND\_PACKAGE( OpenMP REQUIRED)  
if(OPENMP\_FOUND)  
 message("OPENMP FOUND")  
 set(CMAKE\_C\_FLAGS "${CMAKE\_C\_FLAGS} ${OpenMP\_C\_FLAGS}")  
 set(CMAKE\_CXX\_FLAGS "${CMAKE\_CXX\_FLAGS} ${OpenMP\_CXX\_FLAGS}")  
endif()

不同情况下程序的运行时间如下表所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 串行 | 1核 | 2核 | 4核 | 8核 |
| 时间：s | 41.25 | 37.69 | 23.82 | 20.41 | 11.20 |

运行时间截图

Screenshot from 2019-06-20 20-37-19

串行

Screenshot from 2019-06-20 20-30-53

并行化1核

Screenshot from 2019-06-20 20-34-44

并行化2核

Screenshot from 2019-06-20 20-43-06

并行化4核

Screenshot from 2019-06-20 20-41-30

并行化8核

实验结果分析

从上述图标中可以看出，随着线程数目的增多，程序运行时间逐渐降低。

在解决矩阵运算这种大量for循环的问题，通过将for循环进行并行化处理，能够大大的降低程序的运行时间。

**3 Random Reduction**

**环境配置：**

下载MPI包[https://www.open-mpi.org/software/ompi/v3.0/](https://www.open-mpi.org/software/ompi/v3.0/" \t "/home/zdj/Documents\\x/_blank)

之后解压配置环境，这里不再赘述。

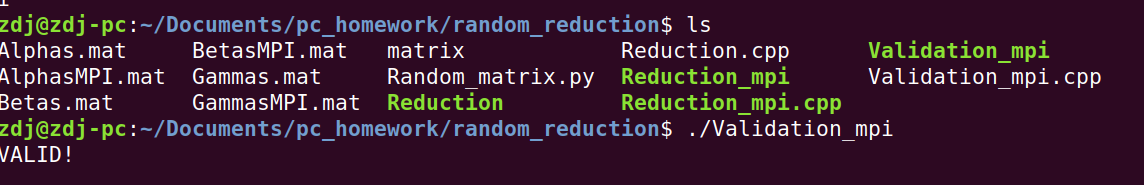
主要命令： mpic++, mpirun

通过python生成23\*23的矩阵（我的学号以及詹姆斯粉丝）

通过Reduction.cpp生成Alphas.mat、Betas.mat、Gammas.mat文件

修改Reduction.cpp文件为Reduction\_mpi.cpp文件并生成AlphasMPI.mat、BetasMPI.mat、GammasMPI.mat文件。

最后运行验证文件得到结果。验证结果截图如下：



4 MPIio

实验要求较多这里不再赘述。

实验结果如图所示

