

# 西北工业大学

# 实验报告

(2020~2021 学年秋季学期)

课程名称 《并行计算》

姓名 严愉程

学号 2017300138

学院 教育实验学院

班级 HC001706

专业 计算机科学与技术

目录

1	实验内容	2
2	实验环境	2
3	实验内容	2
3.1	Monte-Carlo 随机积分算法估算 $\pi$ 值的并行算法	2
3.1.1	算法原理	2
3.1.2	实验步骤	2
3.1.3	运行结果	2
3.1.4	算法分析	2
3.2	Floyd 并行算法	2
3.2.1	算法原理	2
3.2.2	实验步骤	2
3.2.3	运行结果	3
3.2.4	算法分析	3
3.3	N 皇后问题并行算法	3
3.3.1	算法原理	3
3.3.2	运行结果	3
3.3.3	算法分析	3
4	实验总结	3

# 《并行计算》实验报告

严愉程

2020/12/30

## 1 实验内容

1. 熟悉 MPI 编程环境，掌握 MPI 编程基本函数及 MPI 的相关通信函数用法，掌握 MPI 的主从模式及对等模式编程；

2. 熟悉 OpenMP 编程环境，初步掌握基于 OpenMP 的多线程应用程序开发，掌握 OpenMP 相关函数以及数据作用域机制、多线程同步机制等。

## 2 实验环境

ubuntu18.04, intel i7-7700HK 4 核 8 线程, gcc8.2, OpenMPI 4.2.0。使用 CMake 构建。

## 3 实验内容

分别进行以下实验：1. 用 Monte-Carlo 随机积分算法估算  $\pi$  值的并行算法；2. Floyd 算法的并行化；3. N 皇后问题并行算法

### 3.1 Monte-Carlo 随机积分算法估算 $\pi$ 值的并行算法

#### 3.1.1 算法原理

#### 3.1.2 实验步骤

#### 3.1.3 运行结果

#### 3.1.4 算法分析

分析不同  $n$  值、 $P$  值以及不同有序度时算法的运行时间，进行算法并行性能和可扩展性分析。(\*)

### 3.2 Floyd 并行算法

#### 3.2.1 算法原理

#### 3.2.2 实验步骤

第一版的算法，在通讯的手段上比较粗糙，随着最外层变量  $k$  的迭代过程中，每一次矩阵  $mat[][]$  更新都进行一次矩阵的同步，即每个线程都广播一次自己管辖的矩阵区域，如图1。在  $mat\_size =$

4000 的节点数下，多线程的运行时间和单线程的差不多，虽然能够得出正确的结果，但加速比不理想。

```
(base) → build git:(master) * make
Scanning dependencies of target test
[ 14%] Building CXX object CMakeFiles/test.dir/src/floyd.cpp.o
[ 28%] Linking CXX executable test
[ 42%] Built target test
Scanning dependencies of target floyd
[ 57%] Building CXX object CMakeFiles/floyd.dir/src/floyd.cpp.o
[ 71%] Linking CXX executable floyd
[100%] Built target floyd
(base) → build git:(master) * mpirun -n 4 ./floyd
计算时间为: 94.552s
(base) → build git:(master) * mpirun -n 2 ./floyd
计算时间为: 96.7884s
(base) → build git:(master) * mpirun -n 1 ./floyd
计算时间为: 96.7821s
(base) → build git:(master) * ./test
相同
(base) → build git:(master) * 
```

图 1: floyd 代码第一版的运行结果

考虑从通信量上的进行优化。在操作  $mat[i][j] = \min(mat[i][k] + mat[k][j])$  中，最外层循环  $k$  迭代的时候， $mat[i][k]$  的数据永远在本线程管辖的区域内，不需要同步。而  $mat[k][j]$  会随着  $k$  的迭代，出现其他线程管辖的区域或者本线程管辖的区域是不确定的。

所以每次  $k$  迭代的时候，各个线程需要的是  $k$  行数据，迭代  $k$  行之前要广播  $k$  行。

需要注意的是，每次迭代的时候，实际上每个线程管辖的区域都更新了，最后还需要把这些更新进行同步，因为题目要求的是线程依次按照顺序分别写入，就不需要进行同步（在多节点的机器上应该不能做这种写入操作）。

改进之后的运行结果如图2所示。

```
[ 42%] Built target test
Scanning dependencies of target floyd
[ 57%] Building CXX object CMakeFiles/floyd.dir/src/floyd.cpp.o
[ 71%] Linking CXX executable floyd
[100%] Built target floyd
(base) → build git:(master) * mpirun -n 4 ./floyd
计算时间为: 21.1759s
(base) → build git:(master) * mpirun -n 2 ./floyd
计算时间为: 37.5783s
(base) → build git:(master) * mpirun -n 1 ./floyd
计算时间为: 71.6522s
(base) → build git:(master) * ./test
相同
(base) → build git:(master) * 
```

图 2: floyd 优化版命令行运行结果

### 3.2.3 运行结果

### 3.2.4 算法分析

分析不同  $n$  值、 $P$  值以及不同有序度时算法的运行时间，进行算法并行性能和可扩展性分析。(\*)

表 1: floyd 运行结果

矩阵的大小	线程数为 1	线程数为 2	线程数为 4
4000	71.6522s	37.5783s	21.1759s
2000	9.36538s	5.47045s	3.69849s

### 3.3 N 皇后问题并行算法

#### 3.3.1 算法原理

#### 3.3.2 运行结果

#### 3.3.3 算法分析

分析不同  $n$  值、 $P$  值以及不同有序度时算法的运行时间, 进行算法并行性能和可扩展性分析。(\*)

## 4 实验总结