**中山大学计算机院本科生实验报告**

**（2024学年春季学期）**

**课程名称：并行程序设计 批改人：**

|  |  |  |  |
| --- | --- | --- | --- |
| **实验** | **8 -并行多源最短路径搜索** | **专业（方向）** | **计算机科学与技术** |
| **学号** | **21307174** | **姓名** | **刘俊杰** |
| **Email** | **Liujj255@mail2.sysu.edu.cn** | **完成日期** | **2024/5/15** |

# **实验目的**

**1. 使用任意并行框架实现多源最短路径搜索**

**使用OpenMP/Pthreads/MPI中的一种实现无向图上的多源最短路径搜索，并通过实验分析在不同进程数量、数据下该实现的性能。**

|  |  |
| --- | --- |
|  | **IMG_256** |
| **（a）邻接表** | **(b) 图可视化** |

**输入：**

**1. 邻接表文件，其中每行包含两个整型（分别为两个邻接顶点的ID）及一个浮点型数据（顶点间的距离）。上图（a）中为一个邻接表的例子。注意在本次实验中忽略边的方向，都视为无向图处理；邻接表中没有的边，其距离视为无穷大。**

**2. 测试文件，共行，每行包含两个整型（分别为两个邻接顶点的ID）。**

**问题描述：计算所有顶点对之间的最短路径距离。**

**输出：多源最短路径计算所消耗的时间；及个浮点数，每个浮点数为测试数据对应行的顶点对之间的最短距离。**

**要求：使用OpenMP/Pthreads/MPI中的一种实现并行多源最短路径搜索，设置不同线程数量（1-16）通过实验分析程序的并行性能。讨论不同数据（节点数量，平均度数等）及并行方式对性能可能存在的影响。**

# **实验过程和核心代码**

## **2.1 实验思路**

**Floyd-Warshall算法用于计算加权图中所有节点对之间的最短路径。该算法基于动态规划思想，通过中间节点的逐步添加来更新当前最短路径的长度。在单线程版本中，算法的时间复杂度为O(n^3)，其中n是节点数量。为了加速该算法，可以利用多线程技术并行计算不同节点之间的最短路径。**

**2.2 实验过程**

**设计并实现多线程版本的Floyd-Warshall算法： 在本实验中，设计了使用多线程的Floyd-Warshall算法。通过将节点分配给不同的线程，每个线程负责计算部分节点之间的最短路径。这样可以提高算法的运行效率。**

**①生成随机邻接矩阵： 在实验中，生成了一个随机的邻接矩阵作为输入数据。这个邻接矩阵表示了一个加权有向图，其中每个元素代表两个节点之间的距离或权重。**

**②应用多线程Floyd-Warshall算法计算最短路径： 将生成的随机邻接矩阵作为输入，利用设计的多线程版本Floyd-Warshall算法计算所有节点对之间的最短路径。**

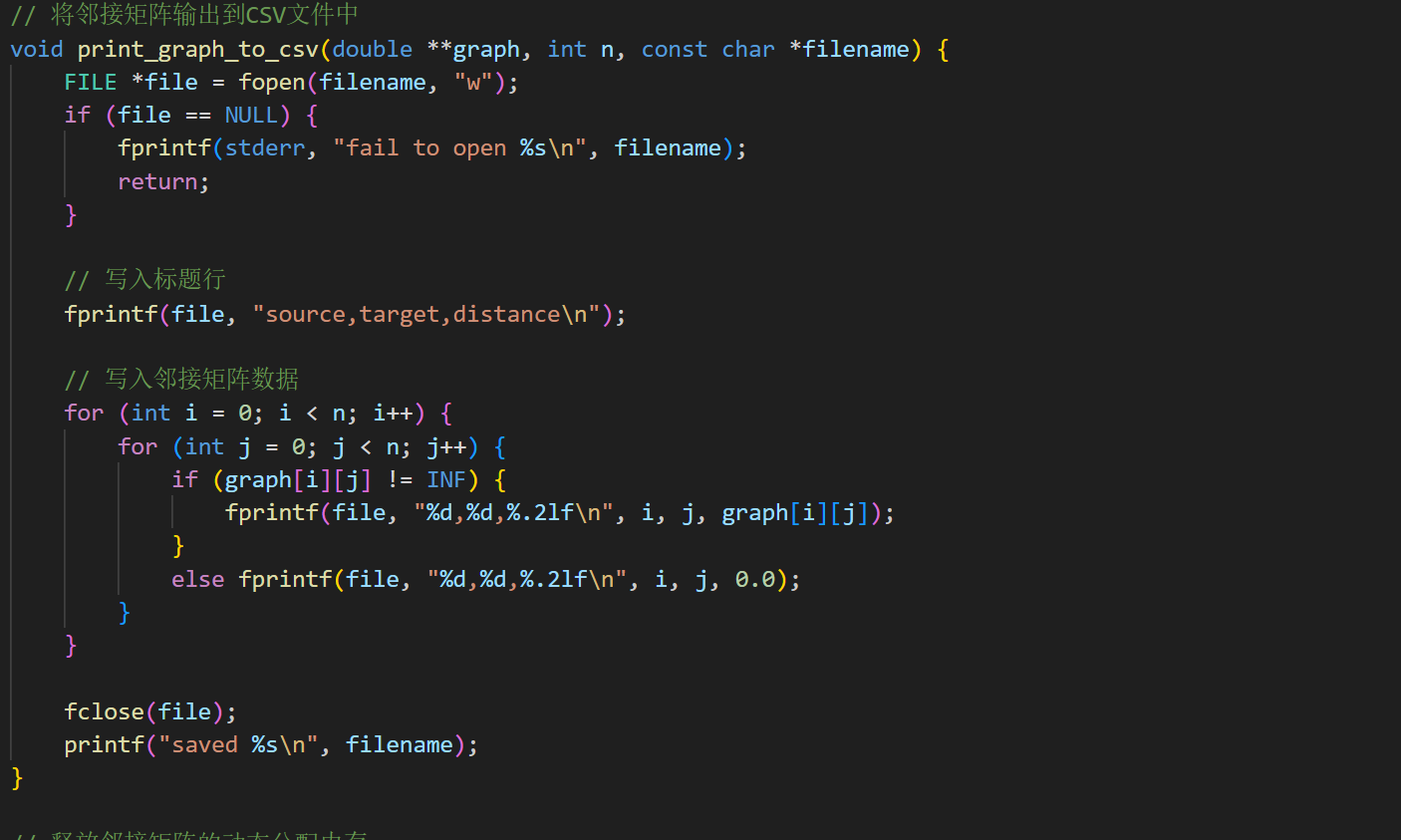
**③将结果输出到CSV文件中： 计算完成后，将得到的最短路径结果输出到CSV文件中，以便后续分析和可视化。**

**2.3 核心代码**

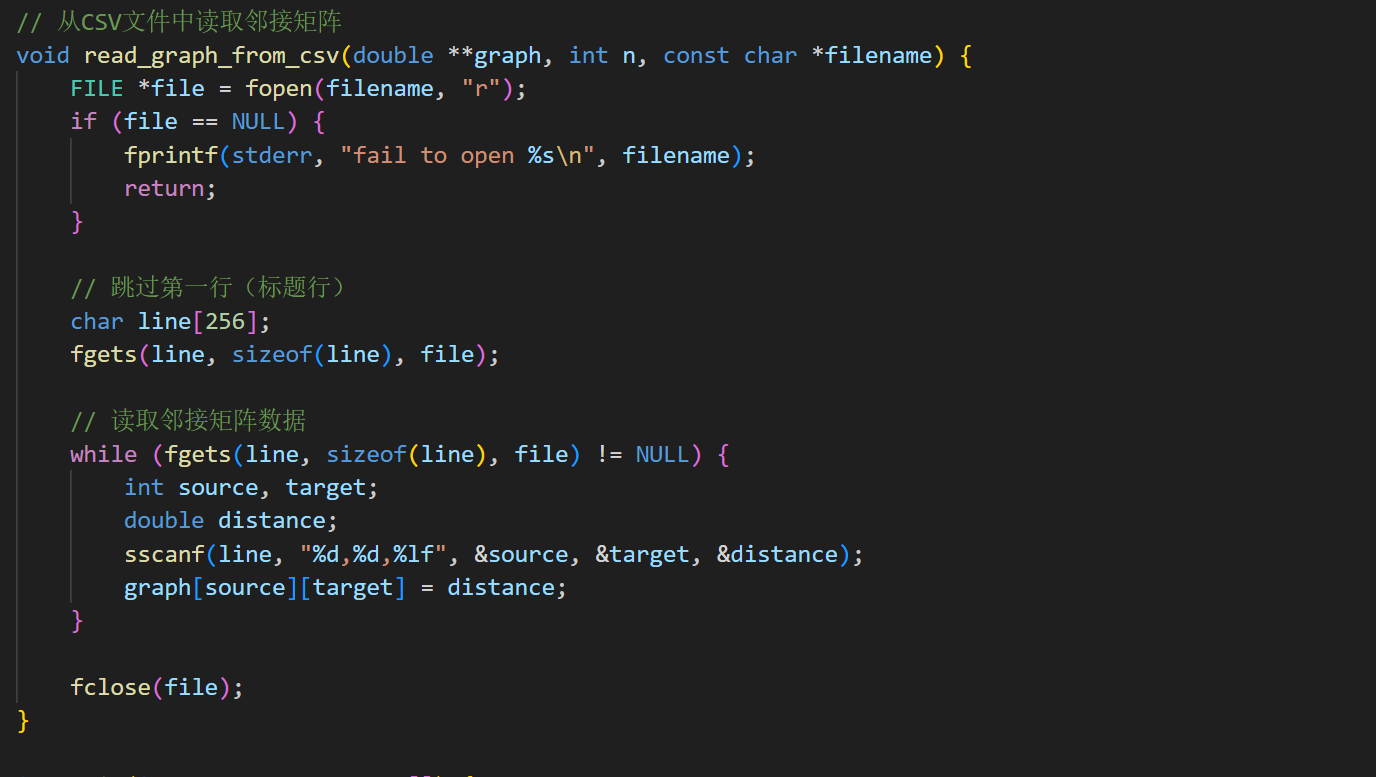
**①生成了一个随机的邻接矩阵作为输入数据。这个邻接矩阵表示了一个加权有向图，其中每个元素代表两个节点之间的距离或权重。**

****

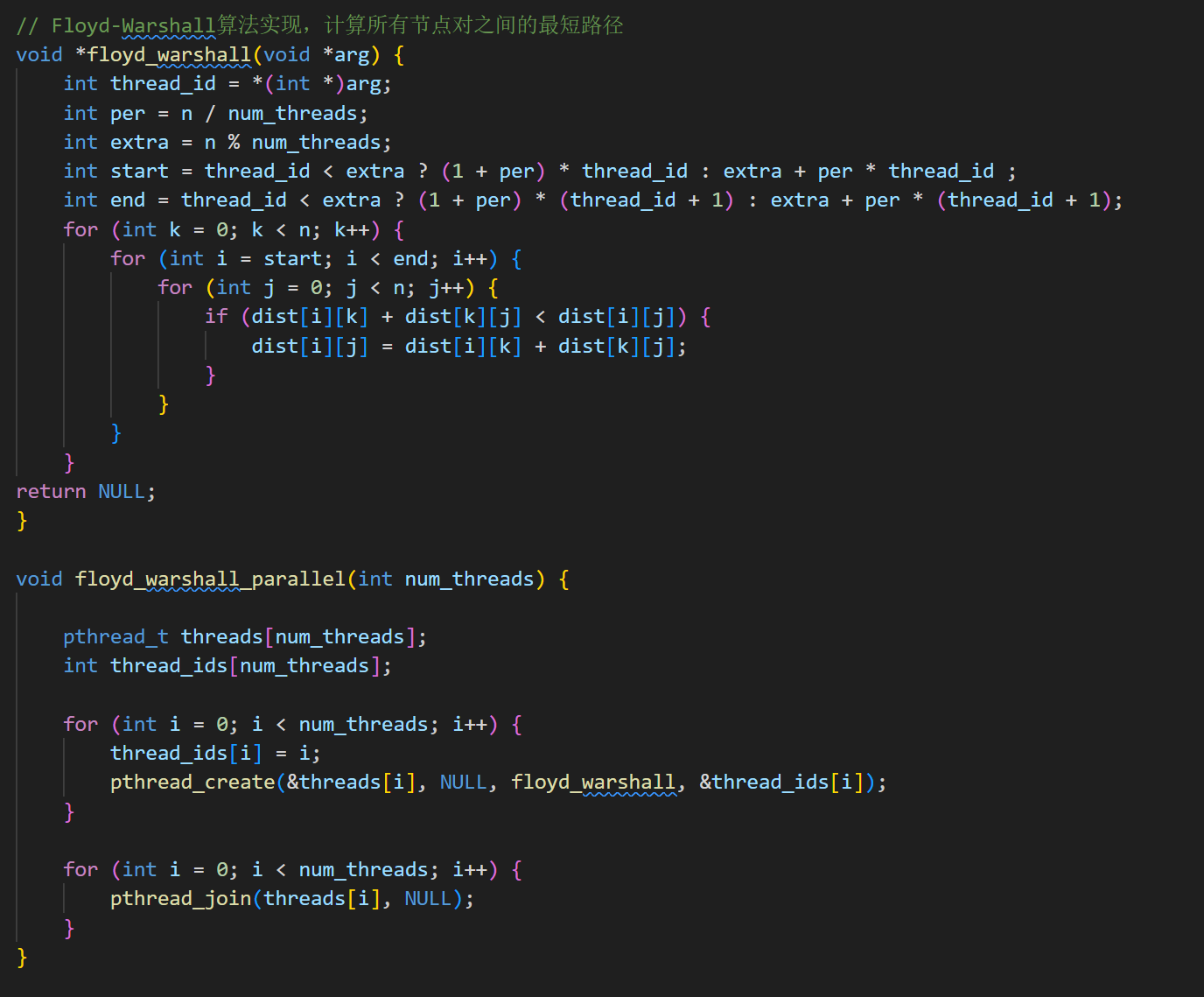
**②将邻接矩阵输出到csv文件中**

****

**③从csv文件中读取邻接矩阵**

****

**④使用pthread多线程实现并行Floyd-Warshall算法**

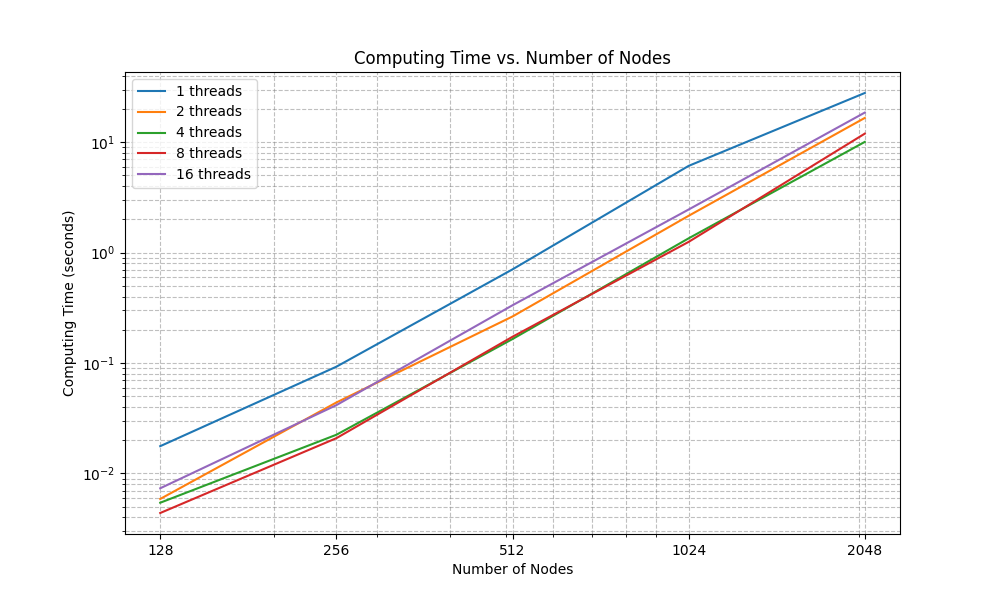
****

# **实验结果**

**3.1 实验结果展示**

## 

**可视化对比:**

****

**3.2 实验结果分析**

**①随着节点数量的增加，计算时间呈指数增长： 无论线程数量如何，随着节点数量从128增加到2048，计算时间都呈现出指数级增长。这表明在Floyd-Warshall算法中，节点数量对计算时间有显著影响，因为节点数量的增加会导致计算量的大幅增加。**

**②多线程并行计算显著提高了计算效率： 在所有节点数量下，随着线程数量的增加，计算时间明显减少。这表明使用多线程并行计算可以有效地利用多核处理器的优势，加快了计算速度。特别是在节点数量较大时，多线程并行计算的优势更加明显。**

**③线程数量对计算效率的影响有限： 尽管增加线程数量可以提高计算效率，但是当线程数量增加到一定程度后，进一步增加线程数量对计算效率的提升有限。在节点数量较小的情况下，增加线程数量可以显著降低计算时间，但是在节点数量较大时，增加线程数量对计算时间的影响逐渐减弱，甚至达到饱和状态。**

1. **实验感想**

**通过本次实验我掌握多线程编程的基本技巧， 在实现多线程版本的Floyd-Warshall算法时，我学会了如何使用pthread库来创建和管理多个线程，以及如何合理地分配任务和同步线程之间的数据访问。这些基本技巧对于编写高效的并行程序非常重要。**