《并行计算》实验报告（摘要）

姓名 学名 完成时间

一、实验名称与内容

二、实验环境的配置参数

（CPU/GPU型号与参数、内存容量与带宽、互联网络参数等）

三、方案设计

（以流程图、框图等方式给出并行方法的思路，简述）

四、实现方法

（结合所用计算环境，给出具体的编程实现方案，简述）

五、结果分析

（在结果正确的前提下，分析所实现方案的加速比、效率等指标，简述）

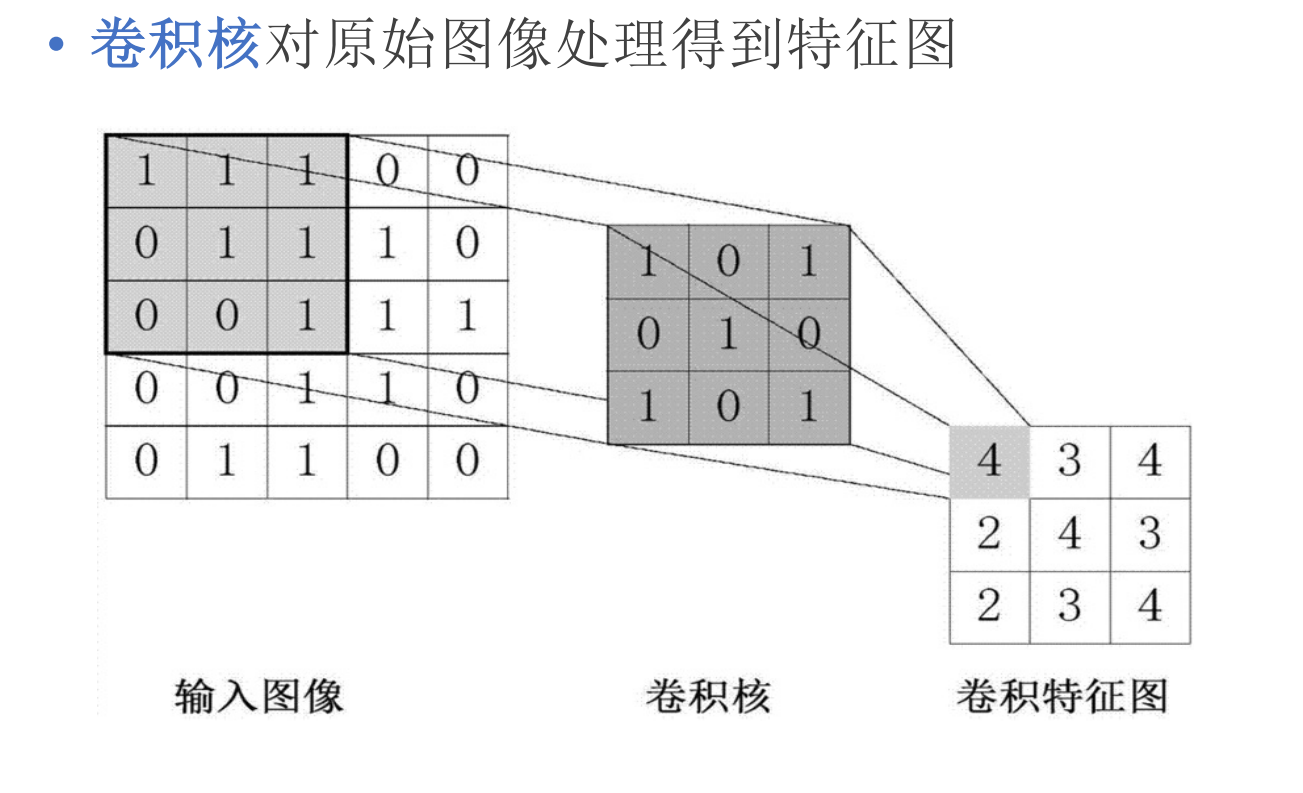
《并行计算》实验报告（正文）

姓名 刘恒星 学名 2022229044 完成时间 2023-4-6

1. 实验名称与内容

实验名称：多线程计算卷积

实验内容：卷积是一种积分变换的数学方法，广泛应用于通信、物理、图像处理等领域。图像处理中，卷积操作就是卷积核（过滤器 / Filter）在原始图像中进行滑动得到特征图的过程，如图所示。



二、实验环境的配置参数

CPU：国产自主 FT2000+@2.30GHz 56cores

节点数：5000

内存：128GB

网络：天河自主高速互联网络 400Gb/s

单核理论性能（双精度）：9.2GFlops

单节点理论性能（双精度）：588.8GFlops

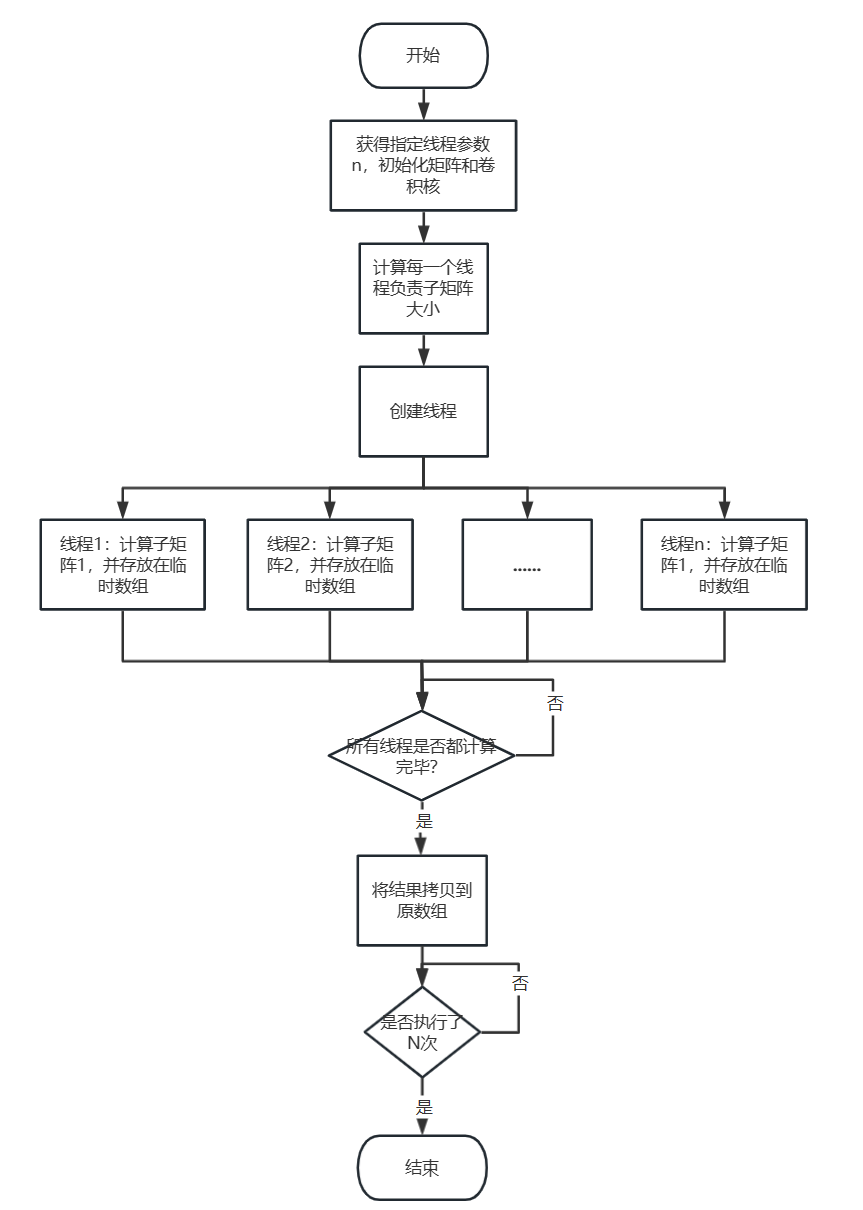
1. 实验题目问题分析

该题目是一个计算矩阵卷积的问题，此问题中，需要卷积核遍历矩阵进行计算。可以抽象为遍历数据域计算最后整合的问题。

对于遍历数据域计算最后进行整合这种类型的问题，我们可以通过划分数据域进行并行优化。具体来说，我们可以讲矩阵划分为子矩阵，每一个子矩阵用一个卷积核进行计算，讲子矩阵的结果保存在临时数组中，最后等待所有线程计算完毕，将数据从临时数据拷贝到原矩阵中，从而达到多线程并行优化的效果。

四、方案设计

流程图如下：



伪代码如下：

conv2d(id):

    start\_row = id \* per\_thread\_rows;

    end\_row = (id + 1) \* per\_thread\_rows;

    result[][];

    for iter from 0 to N:

        for i from 0 to per\_thread\_rows:

            for j from 0 to MAXN - ks + 1:

                sum = 0;

                for ki from 0 to ks:

                    for kj from 0 to ks:

                        sum += filter[ki][kj] \* img[i + ki + start\_row][j + kj];

                result[i][j] = sum;

        pthread\_barrier\_wait(&barrier);

        for i from 0 to per\_thread\_rows:

            for j from 0 to MAXN - ks + 1:

                img[i+1+start\_row][j+1] = result[i][j];

        pthread\_barrier\_wait(&barrier);

for i from 0 to thread\_num:

    ind[i] = i;

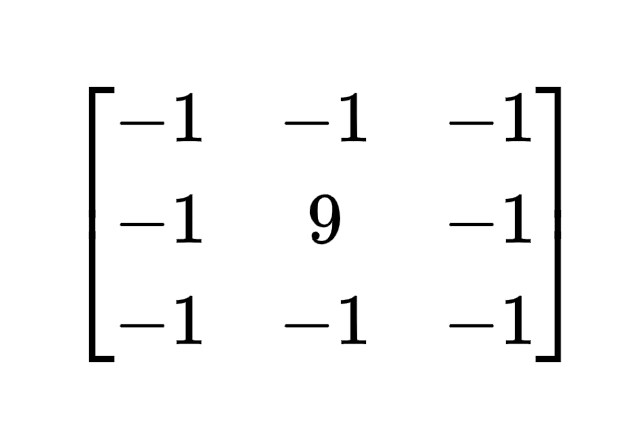
    pthread\_create(&tid[i], NULL, conv2d, (void \*)&(ind[i]));

for i from 0 to thread\_num:

    pthread\_join(tid[i], NULL);

五、实现方法

首先，在程序中定义好矩阵的大小，本次实验定义矩阵原始大小为256\*256，在padding之后大小为258\*258，卷积核大小为3\*3。初始化函数中为原始矩阵中间256\*256的内容填充随机数，卷积核采用的是经典的边缘提取卷积核



。随后从参数中获取线程数，并计算好子矩阵大小。设per\_thread\_row = 256/thread\_num，那么每一个线程负责的子矩阵大小为per\_thread\_row \* 256。

随后进行卷积运算，线程通过函数参数得到id号，从而计算出自己的子矩阵在原始矩阵的起始位置。开辟一个per\_thread\_row \* 256大小的临时数组来记录运算结果。为了防止先计算完成的线程干扰后还在计算的线程，需要等待至所有线程计算完毕之后同意复制结果到原数组。这里使用pthread\_barrier\_wait函数来同步线程。

复制完毕之后，需要用pthread\_barrier\_wait函数等到所有线程都复制完毕，才能进行下一次的计算。

六、结果分析

（在结果正确的前提下，分析所实现方案的加速比、效率等指标）

七、个人总结