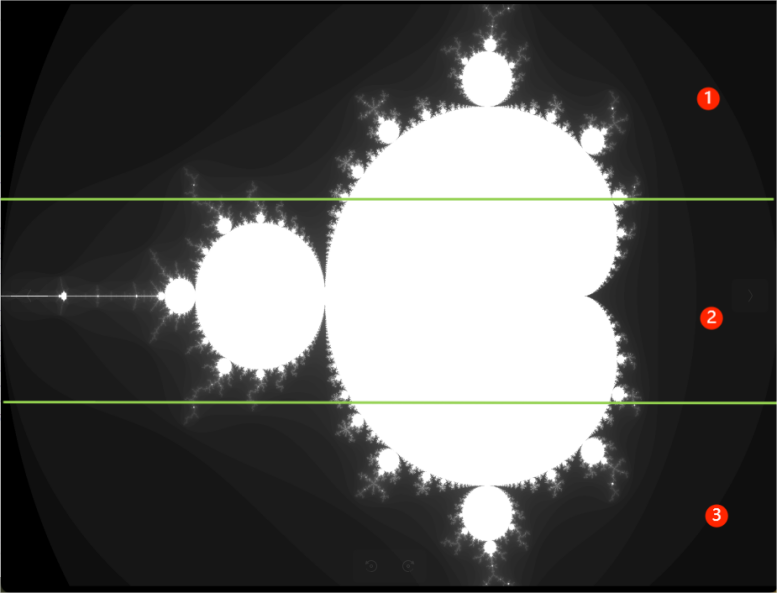
Program 1: Parallel Fractal Generation Using Threads

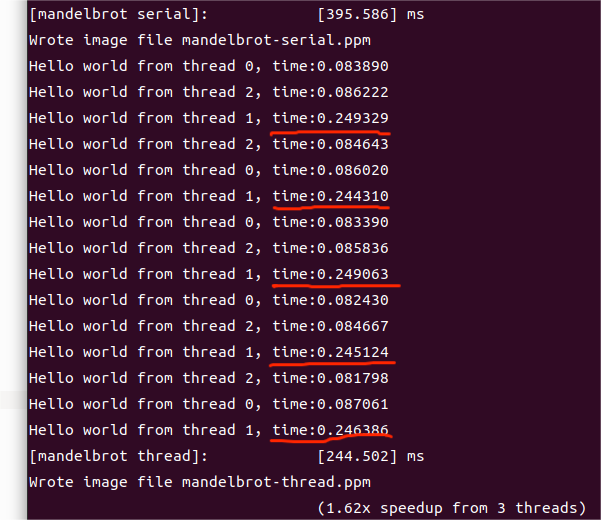
1. 每个线程交替算一行更快。如果分区，例如线程0算上半，线程1算下半，会导致计算区域不均匀，效果较差。采用交叉并行，2线程加速比1.96x。

2.加速比不线性，且线程数为3时反而下降，猜测是因为分块进行时任务分配到每个线程很不均匀导致的。仔细观察图片，中间部分计算量明显偏高。

因此1和3线程计算完后要等待2线程完成，耗时更长。



3.检测每个线程的时间，发现有一个线程确实耗时明显更长，与之前的猜想符合。



4.改善的映射策略：交替并行，例如3线程时，线程1算0、3、6...线程2算1、4、7...线程3算2、5、8... 因为尺寸较大，这样交替划分较为均匀，不会出现负载不均衡的情况。最后8线程时加速比6.73x，明显比按照块划分更好。

5.没有提升，已经达到最大线程数8，多余线程会进入等待，16线程加速比6.06x。

Program 2: Vectorizing Code Using SIMD Intrinsics

1.有关实现见代码文件

2.利用率如下表所示：

|  |  |
| --- | --- |
| vector width | vector utilization |
| 2 | 79.8% |
| 4 | 72.1% |
| 8 | 68.1% |
| 16 | 66.3% |

当向量长度变大时，利用率减小，因为计算时同一组中的值要等待最慢的那一个算完才能保存结果。因此每组都是取最慢的那个数计算的时长，长度越大，越可能变慢。

3.思路：每次读入一个向量，设数组长N，向量长度n，对读入的值连续使用根号n次hadd函数和interleave函数，最后取第一个值作为本向量中所有值的和temp。循环N/n次，每次累加temp即可。

Program 3: Parallel Fractal Generation Using ISPC

1.

2.

3.

4.