作业2 实验报告

1900012731 王一鸣

任务2 并行加速top-down BFS

做法：在取队头结点的循环和进行扩展这两个循环都使用#pragma omp parallel for，并使用#pragma omp critical保护对于new_frontier的更新以及读取，使得new frontier的节点储存正常。

任务3 实现并加速bottom-up BFS

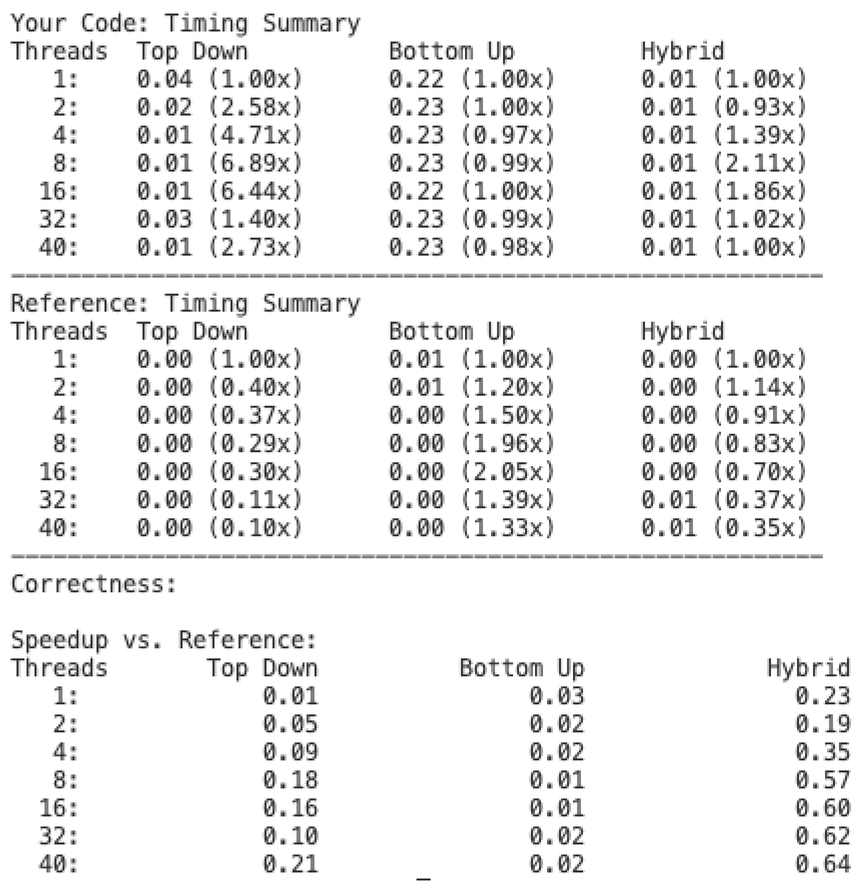
在单线程上成功实现了bottom-up BFS，遗憾的是，它在grid1000x1000.graph以及更大的图上跑不通（超时），因而只能在此报告grid100x100.graph的结果。

加速：对遍历图上结点的大循环做并行加速，并使用critical pragma保护对于new_frontier的更新以及读取。

任务4 实现hybrid BFS

做法：当frontier的size大于图节点个数的四分之一，则采用bottom-up BFS，反之则采用top-down BFS。

测试结果如下，为grid100x100.graph的结果。



```
Your Code: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.04 (1.00x)      0.22 (1.00x)      0.01 (1.00x)
   2:    0.02 (2.58x)      0.23 (1.00x)      0.01 (0.93x)
   4:    0.01 (4.71x)      0.23 (0.97x)      0.01 (1.39x)
   8:    0.01 (6.89x)      0.23 (0.99x)      0.01 (2.11x)
  16:    0.01 (6.44x)      0.22 (1.00x)      0.01 (1.86x)
  32:    0.03 (1.40x)      0.23 (0.99x)      0.01 (1.02x)
  40:    0.01 (2.73x)      0.23 (0.98x)      0.01 (1.00x)
----------------------------------------------------------
Reference: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.00 (1.00x)      0.01 (1.00x)      0.00 (1.00x)
   2:    0.00 (0.40x)      0.01 (1.20x)      0.00 (1.14x)
   4:    0.00 (0.37x)      0.00 (1.50x)      0.00 (0.91x)
   8:    0.00 (0.29x)      0.00 (1.96x)      0.00 (0.83x)
  16:    0.00 (0.30x)      0.00 (2.05x)      0.00 (0.70x)
  32:    0.00 (0.11x)      0.00 (1.39x)      0.01 (0.37x)
  40:    0.00 (0.10x)      0.00 (1.33x)      0.01 (0.35x)
----------------------------------------------------------
Correctness:

Speedup vs. Reference:
Threads        Top Down          Bottom Up             Hybrid
   1:              0.01               0.03               0.23
   2:              0.05               0.02               0.19
   4:              0.09               0.02               0.35
   8:              0.18               0.01               0.57
  16:              0.16               0.01               0.60
  32:              0.10               0.02               0.62
  40:              0.21               0.02               0.64
```

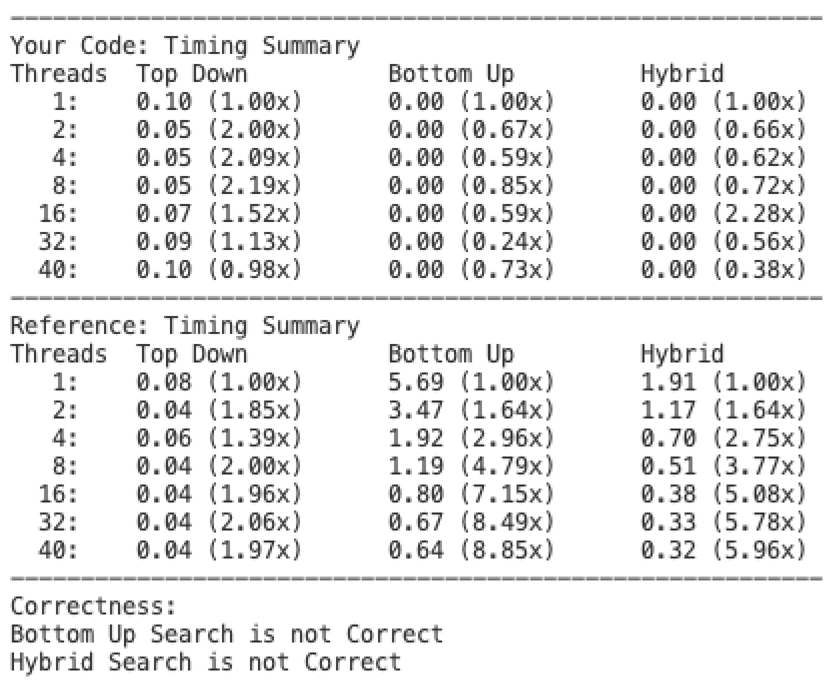
可以看到，并行加速在top-down、hybrid都有效果，并且hybrid的耗时更少。然而，在bottom-up下的BFS，我实现的并行加速并没有明显效果。

分析

- 关于top-down BFS的另一种加速尝试

使用#pragma omp ordered，这样可以省去对于临界区的竞争，缺点是执行顺序固定。

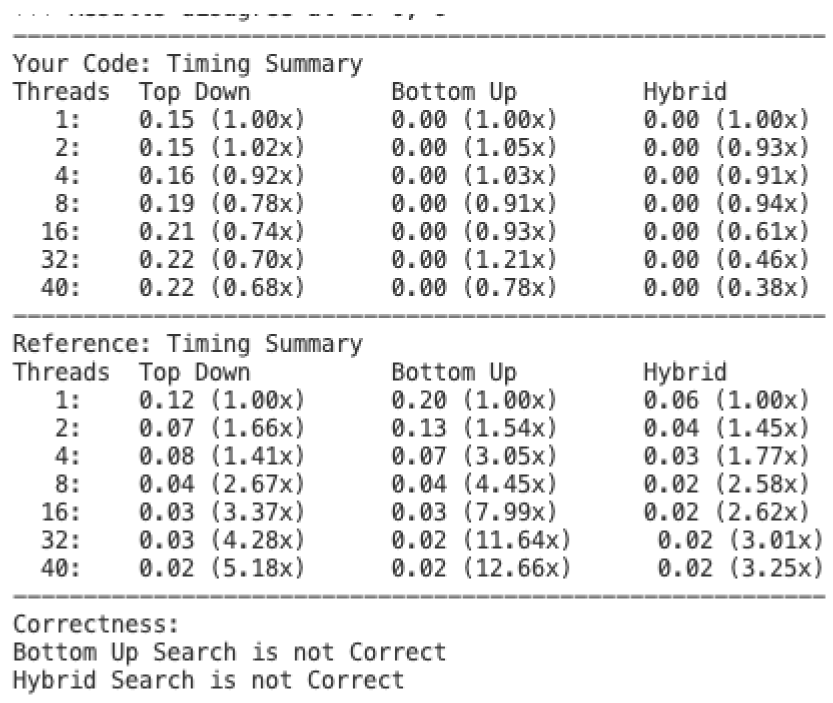
结果如下(在grid1000x1000.graph上测试)：



```
----------------------------------------------------------
Your Code: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.10 (1.00x)      0.00 (1.00x)      0.00 (1.00x)
   2:    0.05 (2.00x)      0.00 (0.67x)      0.00 (0.66x)
   4:    0.05 (2.09x)      0.00 (0.59x)      0.00 (0.62x)
   8:    0.05 (2.19x)      0.00 (0.85x)      0.00 (0.72x)
  16:    0.07 (1.52x)      0.00 (0.59x)      0.00 (2.28x)
  32:    0.09 (1.13x)      0.00 (0.24x)      0.00 (0.56x)
  40:    0.10 (0.98x)      0.00 (0.73x)      0.00 (0.38x)
----------------------------------------------------------
Reference: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.08 (1.00x)      5.69 (1.00x)      1.91 (1.00x)
   2:    0.04 (1.85x)      3.47 (1.64x)      1.17 (1.64x)
   4:    0.06 (1.39x)      1.92 (2.96x)      0.70 (2.75x)
   8:    0.04 (2.00x)      1.19 (4.79x)      0.51 (3.77x)
  16:    0.04 (1.96x)      0.80 (7.15x)      0.38 (5.08x)
  32:    0.04 (2.06x)      0.67 (8.49x)      0.33 (5.78x)
  40:    0.04 (1.97x)      0.64 (8.85x)      0.32 (5.96x)
----------------------------------------------------------
Correctness:
Bottom Up Search is not Correct
Hybrid Search is not Correct
```

可以看到，加速比随线程的增加而减少，因为这样会导致越来越多的线程处于等待状态。

在random_1m.graph上的加速比就更差了：



```
----------------------------------------------------------
Your Code: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.15 (1.00x)      0.00 (1.00x)      0.00 (1.00x)
   2:    0.15 (1.02x)      0.00 (1.05x)      0.00 (0.93x)
   4:    0.16 (0.92x)      0.00 (1.03x)      0.00 (0.91x)
   8:    0.19 (0.78x)      0.00 (0.91x)      0.00 (0.94x)
  16:    0.21 (0.74x)      0.00 (0.93x)      0.00 (0.61x)
  32:    0.22 (0.70x)      0.00 (1.21x)      0.00 (0.46x)
  40:    0.22 (0.68x)      0.00 (0.78x)      0.00 (0.38x)
----------------------------------------------------------
Reference: Timing Summary
Threads  Top Down          Bottom Up         Hybrid
   1:    0.12 (1.00x)      0.20 (1.00x)      0.06 (1.00x)
   2:    0.07 (1.66x)      0.13 (1.54x)      0.04 (1.45x)
   4:    0.08 (1.41x)      0.07 (3.05x)      0.03 (1.77x)
   8:    0.04 (2.67x)      0.04 (4.45x)      0.02 (2.58x)
  16:    0.03 (3.37x)      0.03 (7.99x)      0.02 (2.62x)
  32:    0.03 (4.28x)      0.02 (11.64x)      0.02 (3.01x)
  40:    0.02 (5.18x)      0.02 (12.66x)      0.02 (3.25x)
----------------------------------------------------------
Correctness:
Bottom Up Search is not Correct
Hybrid Search is not Correct
```

不过，其运行时间还是比较优的。

- 关于bottom-up BFS的运行失败。

个人猜测是因为算法本身效率过低，当然也可能是我的实现(按照提供的伪代码实现)有可以优化的地方。