Parallel Solution of the Heat Distribution Problem

Yonggang Li (18043158)

**Abstract**

This report is about an OpepMP parallel solution of the heat distribution of a printed circuit plate. The architecture of the operating system, the design of the program, the performance comparison of running the program on different number of processors and different sizes of the printed circuit plate are introduced. The heat distribution pictures are also attached.

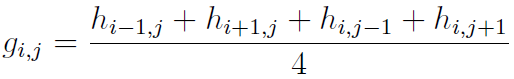
1. **Workstation**

The program is built and tested on a workstation in the computer lab of Massey University, details as below.

* Processor Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz, 3408 Mhz, 4 Core(s), 4 Logical Processor(s)
* Operating system Windows 10.0.16299
* Physical memory: 8 GB
* OpenMP version xxx
* Edit and Compiled with Microsoft Visual C++ 2017

1. **Heat distribution formula and data structure**

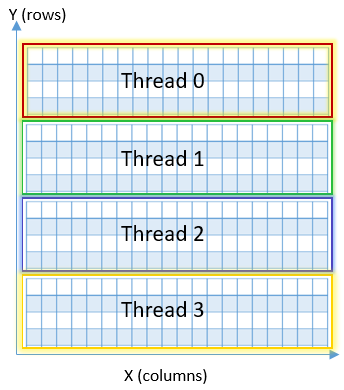
The updated temperature then depends upon the temperature of the points immediately above and below and to the left and right:



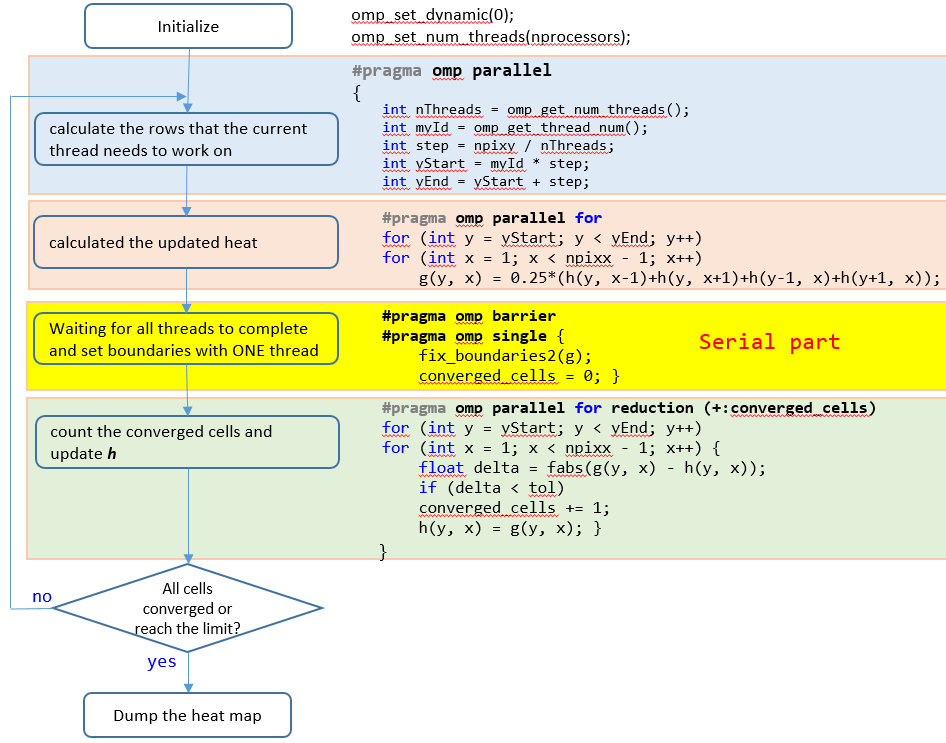
There are two identical 2-dimension arrays, ***h*** and ***g***, to store the heat of the plate. ***g*** is as a temporary cache for each round to store the updated heat values and the heat values will be copied to ***h*** after the convergence computing finishes.

1. **Parallel strategy**

Let’s assume there are N processors. The plate is split into the N parts and the same number of threads will be created. Each thread will calculate the heat distribution of one part. When all threads completed their calculation, a single thread will work on setting the boundaries on ***g.***



Then, threads are working on computing the difference of the corresponding cells of ***h*** and ***g*** one by one and update ***h*** with ***g***. If the absolute value of the difference is small enough, less than 0.00001 in this solution, we say it is “converged”. The computing will end if all cells are converged, otherwise, repeat the calculation until all cells are converged or the limit of rounds is reached. The figure below show the whole process:



1. **Testing result**

The first test is to let each processor handle about 10,000 pixels, while increase the plate size and the number of processors at the same time. Table 1 shows the result.

|  |  |  |  |
| --- | --- | --- | --- |
| Plate size | # processors | Total time | Parallel part time |
| 100 | 1 | 0.073252 | 0.066843 |
| 200 | 2 | 0.087856 | 0.069855 |
| 400 | 4 | 0.128181 | 0.078841 |

Table 1 Total and parallel time for different plate size on different processors

As Gustafson’s law ,where f is the fraction of parallelized part of the total work,

The second test is calculate a fixed size of plate with different number of processors.

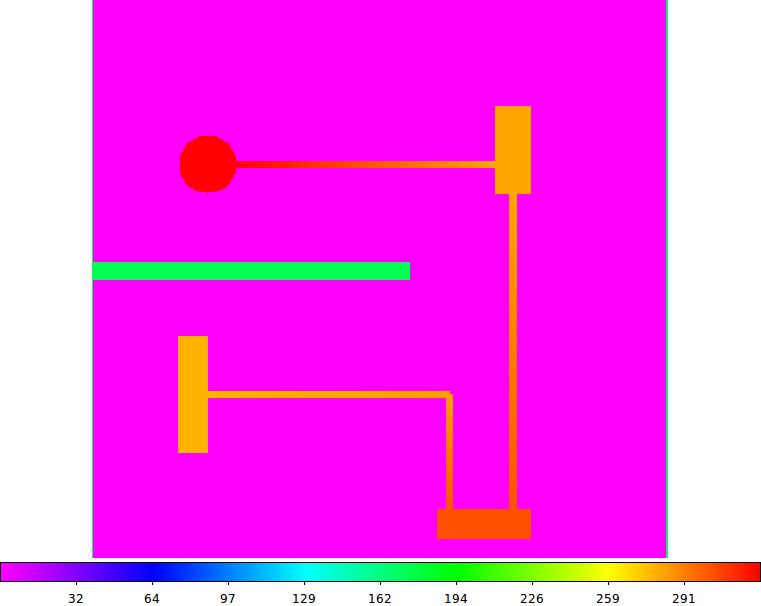
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N  (processors) | Total  time(s) | TS (time of serial part) (s) | F  TS/Total | speed-ups | |
| Theoretical | observed |
| 1 | 0.092889 | 0 | 0 | 1 |  |
| 2 | 0.047515 | 0.00004 | 0.000841839 | 1.99992 | 1.95 |
| 4 | 0.023604 | 0.000211 | 0.008939163 | 3.99747 | 3.94 |

Table 2: Time and speedups for plate size 400

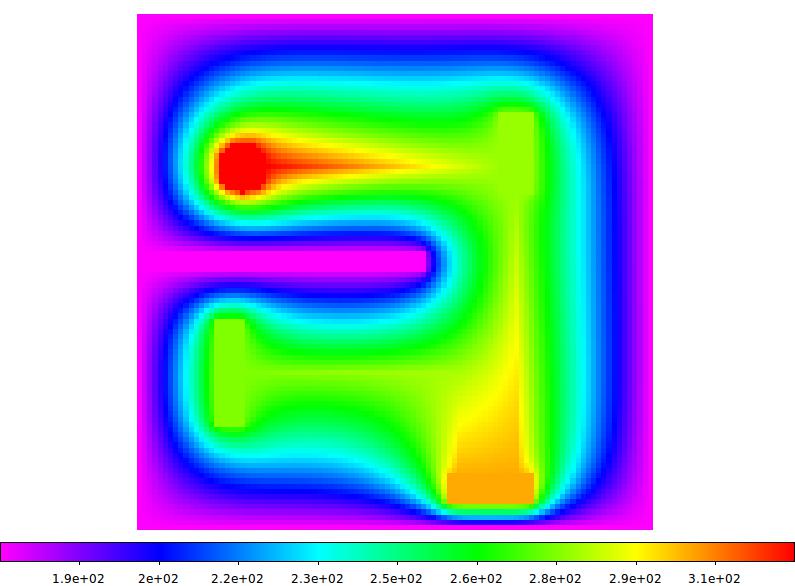
\* Observed Speedups = Total time of one processor / Total time of N processors

1. **Result Heat Map**

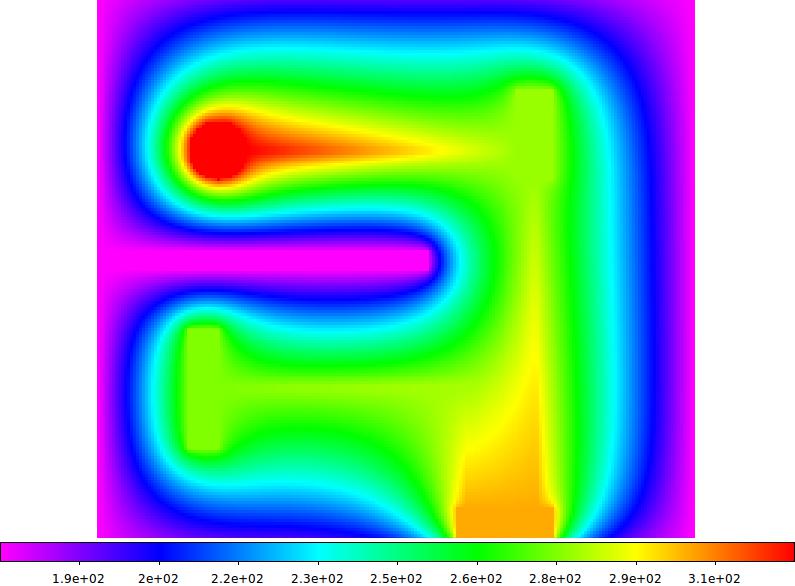
Only one image for initial plate is attached as Picture 1, which is 400x400. Picture 2, 3 and 4 are the converged plates for 100x100, 200x200, and 400x400 respectively. We can see as the size increase; the image is becoming smoother.



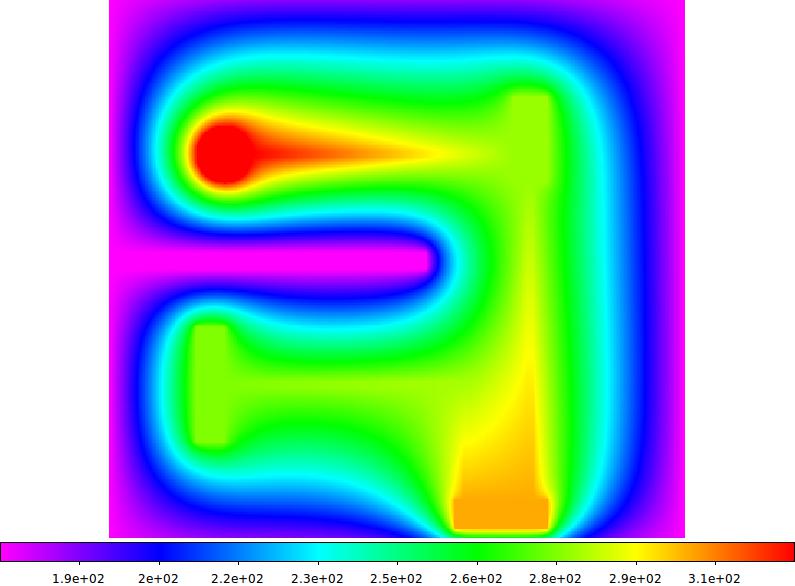
Picture 1 Initial Heat Map for 400x400



Picture 2 Converged Heat Map for 100x100



Picture 3 Converged Heat Map for 200x200



Picture 4 Converged Heat Map for 400x400

1. **Conclusion**

The test result demonstrates Gustafson’s Law that the more processors can calculate the bigger size of the circuit plate without affecting the performance. On the other hand, given a specific number of processors, as we lower the serial fraction parts of the total work, we gain performance increase, this also conform to Amdahl’s law.