# APC 524 / AST 506 / MAE 506 Software engineering for scientific computing
# Assignment 4: Parallel programming with OpenMP and MPI

Hao Zhang, MAE, haozhang@princeton.edu

November 19, 2016

## 1 Computation time

For the sake of time, we only consider grid size $nx = 64, 128, 256$. It's found that OpenMP and MPI produce nearly-linear speed-up. With the increase of threads, the perfect linear speed-up is actually limited by process that can not be (easily) parallelized(i.e., I/O process).

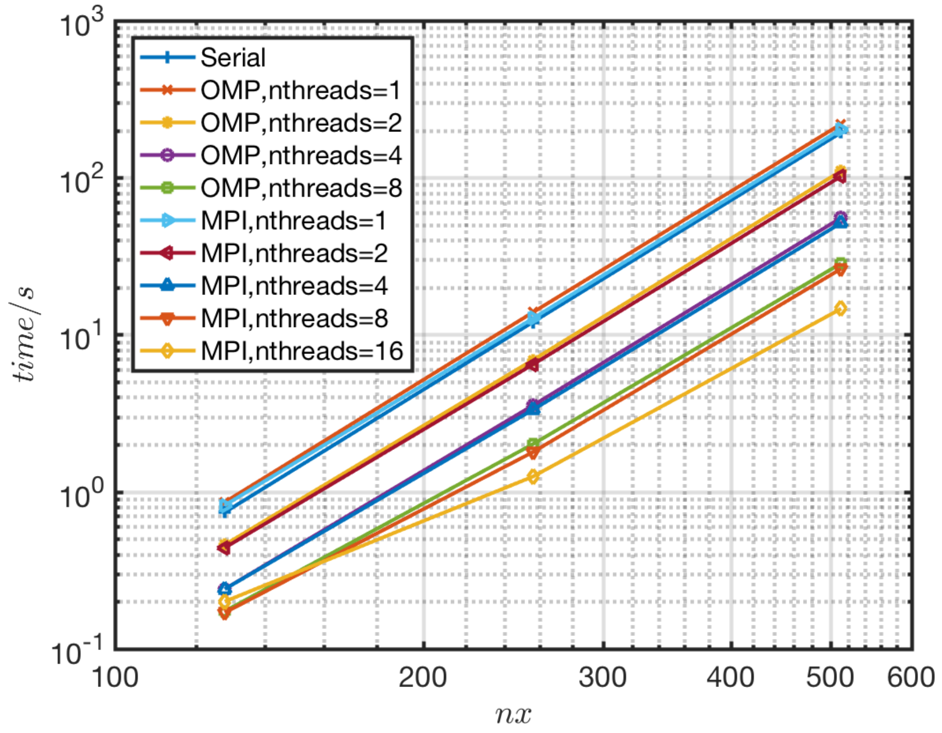

Figure 1: Computation time of serial, OpenMP, MPI (with various threads) with respect to grid size $nx = 64, 128, 256$. All simulation is run on Adroit cluster.

Average temperature is not reported here, but they are almost the same ($T_{mean} \approx 0.4930$) for all runs.

## 2 Single thread, Serial vs OpenMP vs MPI

With a single thread, serial, OpenMP, and MPI takes almot the same computation time. The temperature field are visually identical.

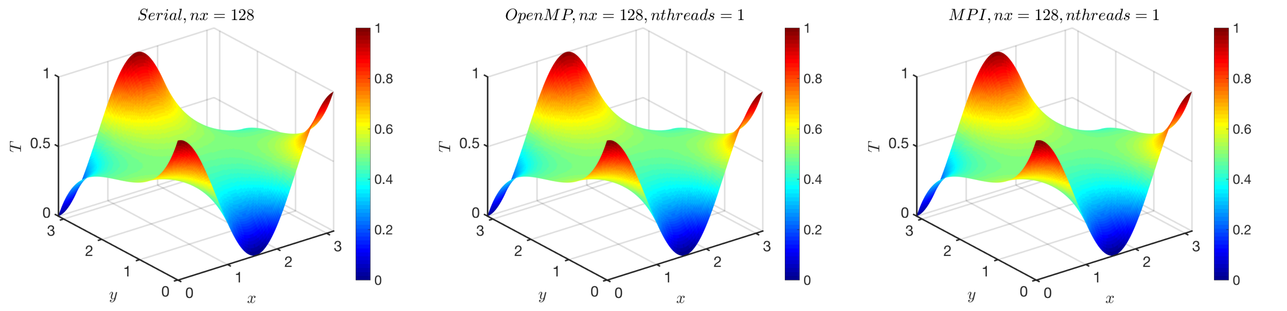

Figure 2: Temperature field computed by serial, OpenMP, and MPI with a single thread, grid size 128 × 128

## 3 OpenMP, multiple threads

To parallel serial code with OpenMP, we only need to add a few new lines before the for loop. A header file ”omp.h” also needs to be included. It’s extremely easy to implement, and the gain is large(depending on the number of threads in the processor).

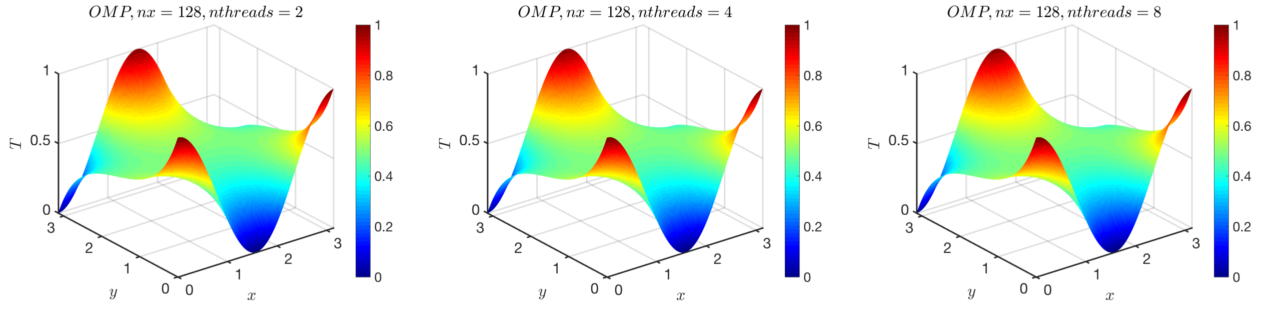

Figure 3: Temperature field computed by OpenMP with nthreads = 2,4,8, grid size 128 × 128

## 4 MPI, multiple threads

When MPI is used to parallelize the updating step, domain decomposition is ultilized to divide tasks into small chunks. The boundary points are passed between various threads, by calling function MPI-Send() and MPI-Recv().

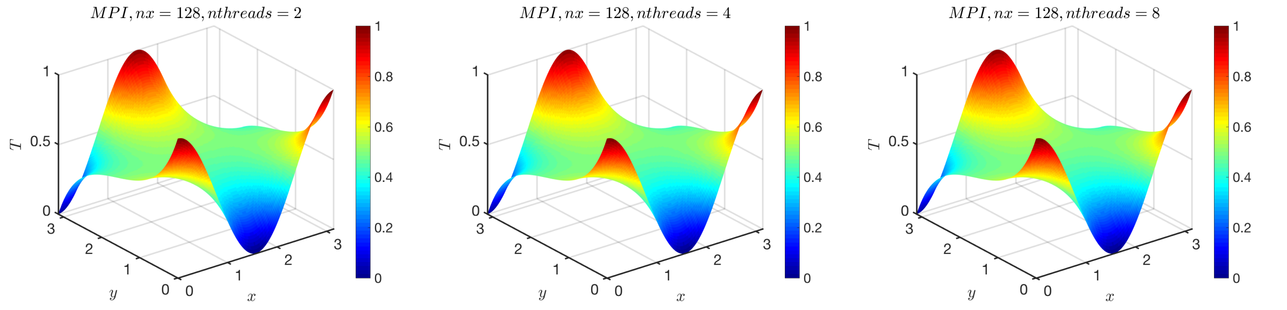

Figure 4: Temperature field computed by MPI with nthreads = 2,4,8, grid size 128 × 128

## 5 OpenMP vs MPI

OpenMP is very easy to use, we only need to add a few lines to the serial code and include ”omp.h” header file. However, OpenMP only works on a single processor, so speed-up is limited by the number of threads available.

MPI takes more effort to implement, and it’s more error-prone. However, MPI in principle works for much more systems/hardwares architecture. For example, it can make full use all the processors on one machine, and even combine the computation power of multiple machines.