Low-Level Parallel Programming
Uppsala University – Spring 2015
Report for Lab 1 by Team 21

Markus Palacios, Jonathan Sharyari

February 5, 2015

1 Questions

A What kind of parallelism is exposed in the identified method?

We have used task parallelism in the identified method. The vector containing the agents is been split up into the same number of processors of the running system, where each sub-vector is individually traversed by the deployed threads. As the same task is divided upon several threads, this is a type of data parallelism.

B How is the workload distributed across the threads?

The workload is distributed evenly across the threads. For Pthreads, this means each thread gets equally many consecutive agents. Using OpenMP, the distribution is automatic, but using the standard settings, the same approach will be taken as with pthreads.

C Which number of thread gives you the best results? Why?

Our data suggests that the number of threads should generally be equal to the amount of processors in the system. In our case, the running system employs two virtual cores (SMT), which seems to benefit from the use of added threads, up to four, in the case of $OpenMP$ parallelisation (possibly due to hyperthreading). We believe the cause of $Pthreads'$ decline in performance is due to the greater overhead suffered.

D Which version (OpenMP, Pthreads) gives you better results? Why?

As we can see in the figure 1, our graph shows that we get a much better result from using OpenMP as compared to Pthreads. As mentioned before this may be due to the larger overhead in Pthreads. When using Pthreads, it is up to the programmer to distribute the workload to the individual threads. We believe that the overheads of using Pthreads, including creating and copying the in-data is too large. This could be overcome by moving the parallelism up in the code hierarchy, i.e. instead of creating a new set threads each tick, they could be instantiated only once, then continuously maintain a portion of the agents.

2 How to run

Without an argument specifying the type of parallelism, the serial version is used, whereas –pthreads and –openmp activates pthreads and openmp respectively (in case both are set, the last occurance will be dominating). The number of threads can in both cases be set by –np X. The value of X is ignored in the serial case.

Serial ./demo

OpenMP ./demo –openmp

Pthreads ./demo –pthreads –np 4

A Experiments

Our experiments were run under Ubuntu Linux 14.04 (64 bit) on an Intel Core i3 550 of 3.2 GHz with an 4 MB L2 cache and a 4 GB RAM.

Intellectual Property

We certify that this report is solely produced by us, except where explicitly stated otherwise and clearly referenced, and that we can individually explain any part of it at the moment of submitting this report.

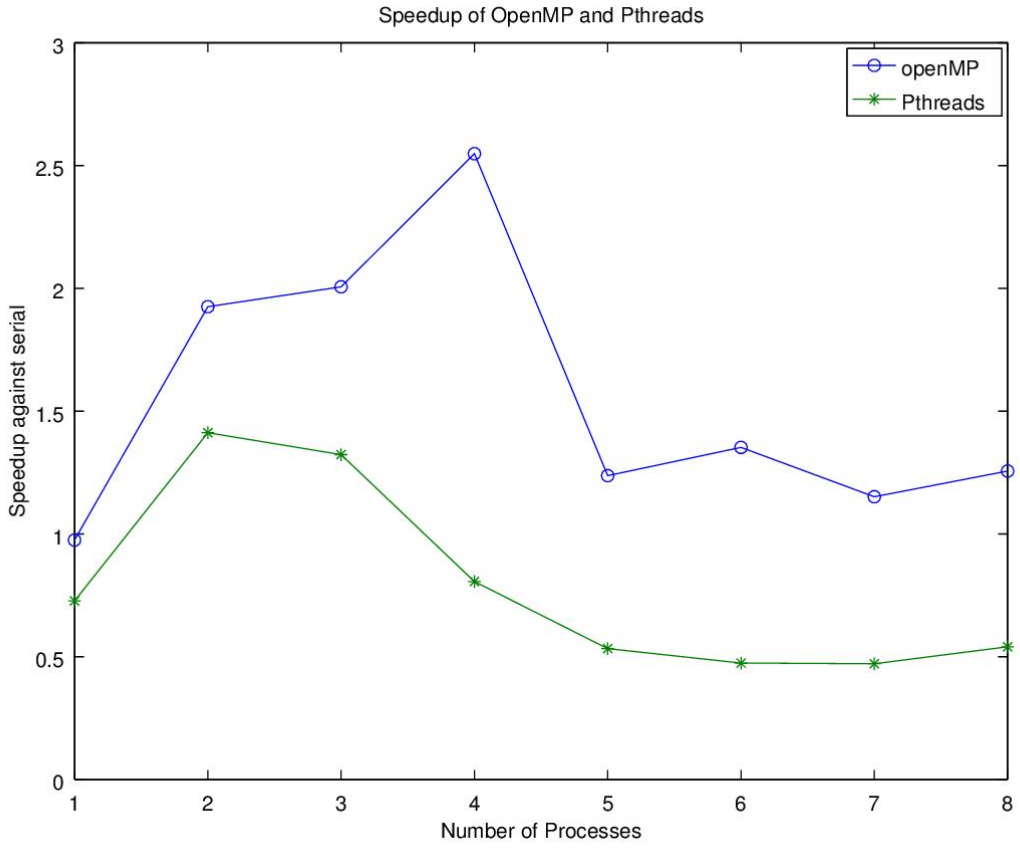

Figure 1: Graph showing the speedup of openmp and pthreads for 1 to 8 threads.