ECE445: PARALLEL AND DISTRIBUTED COMPUTING
Fall 2021-2022

<u>**Homework 3:**</u>
<u>**MPI Programming**</u>
(*Deadline: Friday 11.02.2022, 23:59*)

<u>General Instructions:</u>
Follow the template text (template.docx) since this is part of your grade.
Write clearly your names and your IDs on the first page.
In the theoretical exercises, write in detail your answers and the sources from which you drew them.
In the programming ones, present and explain the algorithm you programmed and present in detail the experimental results of your programs.
Describe your implementation and the machine you worked on (hardware features, OS, compiler, etc.).
Program in C and Linux, use Makefile to compile / link / execute your programs.
Report your observations and present the results with tables and graphs.

Graphs and figures must be software-generated and not handmade and scanned.

In general, your text should be well written and legible, while you should FULLY justify the steps you followed, and comment on the results of each exercise. (20 credits)

Submit your work via eclass to a zip file (hw3_id1_id2_id3.zip), which will contain the text with your solutions / results and comments, your code (.c, .h files and your Makefile).

It is noted that the teams do not change during the semester.

Example-programs with MPI code are available in the .zip at eclass folder « Έγγραφα → Αρχικός Κατάλογος → Open MPI material → MPI_examples.zip »

Exercise 1.
Write a program in C (ask1.c) that implements parallel programming with Open MPI. Your program should
a) (5) display on the screen the number of tasks participating in the execution
b) (5) display on the screen the message: «*Hello. I am the master node.* » from the master task. Afterwards this message will be sent (by the master task) to all tasks.
c) (5) as soon as the other tasks receive the above message, they will respond by displaying on the screen the message «*Hello. This is node *.*» where * is their id. After that, they send the particular message to the main task.
d) (5) the main task will confirm that all the tasks have been answered with an appropriate message on the screen (e.g., «* nodes replied back to master node.» where * the number of tasks that sent their message).
e) (10) the main task will count the execution time of the broadcast and will display it on the screen.
f) (10) Add a table in your report, which will show the number of tasks and the corresponding execution time on your computer.

Number of tasks	1	2	4	8	16	32	64
Time (sec)							

Table 1 : Broadcast execution time.

Exercise 2.
Write a program in C (ask2.c) that implements parallel programming with Open MPI and calculates an approximation of π, using the formula

$$\pi = \int_0^1 \frac{4}{1+x^2}\, dx$$

Use Simpson's rule to approach the integral, i.e.,
if x_i = i*dx, i=0,1,…,N, dx= 1/N, a partition of [0,1] with N intervals (N + 1 points)

$$\pi = \int_0^1 \frac{4}{1+x^2}\, dx \approx \frac{dx}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + \cdots + 2y_{N-2} + 4y_{N-1} + y_N)$$

where $y_i = \frac{4}{1+x_i^2}$, $\mathrm{i} = 0,1,2,\dots,\mathrm{N}$.
Your program should compute correctly the above sum by:
a) (10) The main task should compute all the x_i points and distributes them appropriately to the other tasks
b) (10) The sum will break into nt (= number of tasks) partial sums and each one for a particular task.
c) (20) The assignment of the terms of the sum to the tasks will be done in blocks according to the number of tasks. That is, the task with id = 0, will add the terms with κ = 0, 1, 2,…, ((N + 1) / nt) -1, where nt the number of tasks, the task with id = 1 will add them terms with κ = (N + 1) / nt,…, (2 (N + 1) / nt)-1, etc.
The total sum will be computed by the main task and will be displayed on the screen accordingly.
d) (10) Measure the execution time of calculations, the communication time and calculate the number of operations per sec.
This message should appear on the screen:
pi is approximately * , computations time = *, communication time = *, number of tasks = *, FLOPS = *

e) (10) Add a table in your report that will contain all the above information.
Hint: FLOPS = number of operations per sec. Use N+1 = 10^8 , 10^{10}, 10^{12}, 10^{15} … and nt = 1, 2, 4, 8, 16, 32 ….

Number of tasks	1	2	4	8	16	32	64
Computations Time (sec)							
Communication Time (sec)							

Table 2 : Computation and communication time vs number of tasks while calculating π.

Exercise 3.
Write a program in C (ask3.c) using Open MPI, which implements the Jacobi method for system solution

$$Ax = b \iff \begin{bmatrix} 2 & -1 & \\ -1 & \ddots & -1 \\ & -1 & 2 \end{bmatrix} x = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ N+1 \end{bmatrix}$$

where N is the dimension of the system.
Read slides 8-11 of 02_parallel_algorithms.pdf on the course website (eclass → documents) for more information on the Jacobi method.
Your program should:
a) (20) run for m iterations
b) (40) in each iteration it should calculate the new approach of the solution, xnew, (and display on the screen) the remainder of the approach, i.e., $|| \text{b - A xnew} ||_2$, and the difference of the last 2 approaches, i.e. $|| \text{xold - xnew} ||_2$, with appropriate comments, e.g.:

iter = *, residual = *, difference = *

c) (20) also display the execution and communication time (till the very last iteration)

computations time = *, communication time = *

d) (20) Run your program for different number of threads (1, 2, 4, 8, 16, 32) and add a table to your report, which will show the number of tasks and the corresponding execution time on your computer.

Hint: The system solution is the vector [1 2 3… N]. Use $N = 2^{10}$, m = 50 and nt=1,2,4,8,16,32,…

Number of tasks	1	2	4	8	16	32	64
Computations Time (sec)							
Communication Time (sec)							

Table 3 : Computations and communication time vs number of tasks for the Jacobi problem.

Instructions for MPI on PC/laptop:

Download and install Open MPI (https://www.open-mpi.org/) on your PC/laptop.