This assignment is based on the matrix multiplication project (Project 2) of Chapter 4. Implement a parallelized matrix multiplication program, once using POSIX threads, and once using conventional processes with shared memory (fork() & shmget()). Conduct a series of experiments to record the turnaround times and memory usage, and compare the efficiency of processes vs threads. Use the following test cases:

a. 256 x 256 matrix size, with 4, 16, 64 and 256 threads/processes
b. 512 x 512 matrix size, with 4, 16, 64 and 256 threads/processes
c. 1024 x 1024 matrix size, with 4, 16, 64 and 256 threads/processes
d. 2048 x 2048 matrix size, with 4, 16, 64 and 256 threads/processes

Once you complete the experiments, answer the following questions:

1. [30 pts] Design and implement a methodology of the experiments for measuring the turnaround time.
2. [30 pts] Design and provide the proper graphs/charts that demonstrate the turnaround time, and how the time increases with respect to matrix size and with number of threads.
3. [30 pts] From the experiments conducted, is there an ideal number of threads with respect to matrix size? Does this ideal mapping follow a function, and, if so, what is this function? What can you tell about the number of processes with respect to the matrix size?
4. [30 pts] What can you tell about the scalability of each of the two approaches (threads vs processes) with respect to matrix size and with respect to number of threads/processes?

Notes: do not forget to take into consideration the number of processors or cores on the tested system. Use the graphs of question 1 to answer questions 2 and 3. In your report, include the complete source code of both programs. Source codes should be well commented and properly formatted in fixed-width (a.k.a. monospaced) font; pay particular attention to taking care of lines that exceed 80 characters in length.

Review criteria

1. Does the report describe the measuring methodology?
2. Is the methodology described adequate for the purpose of the experiments? Are the results statistically stable? Do they report the error or standard deviation?
3. Are results reported for all cases? Are they summarized in a table?
4. Does each chart provided serve a specific purpose and have a clear and distinct message from the others?
5. Are the charts readable? Are they properly placed, numbered and captioned?
6. Does the report argue on a trend on the “optimal” number of threads and processes with respect to the matrix size? Is the argument supported by the charts? Does the report provide an analytical formula that is supported by the results?
7. Does the report reach a definite conclusion about the scalability? If not, does it argue what it takes to reach a conclusion about the scalability?
8. Is the code provided? Is it readable, well commented and properly formatted (e.g. line breaks)?
9. Is the report well structured and sectioned, including introduction, problem statement, description of the approach and a description of the results?
10. Does the report use a clear and impartial language with correct grammar and punctuation and full sentences?
11. Does the report properly reference/cite any external sources and material used in the work?
12. Does the report feature a self-evaluation section of the author?