Operating Systems

Report - Assignment 2

Multi-Process & Multi-Threaded Computation of Statistics

CS17BTECH11050 - SHRAIYSH VAISHAY

Goal

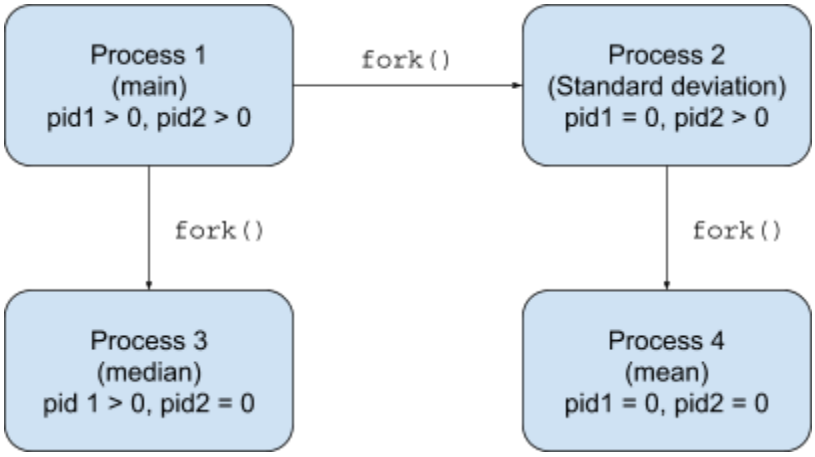
The goal of this assignment is to compute some statistics over a sequence of input numbers using multiple threads, processes and compare the times taken by each of them.

Low level design

Both the programs follow the same basic rule. The main task is created and then the work is divided in different tasks - one for each, mean, median and sample standard deviation.

proc-stat.cpp

- When the program starts, the main process (Process 1) takes the input and stores it in heap. Then it spawns a process for calculation of standard deviation (Process 2).
- The main process starts another process for calculation of median (Process 3), while standard deviation (Process 2) starts another process for calculation of mean.



- Each worker process creates its own shared memory, which it shares with the main process (Process 1) - with the worker being the producer and main (Process 1) being the consumer.
- Process 2 - creates Process 4 and *starts calculating mean of X^2, while Process 4 is calculating mean of X*. Once the computation of $\overline{X^2}$, and $\overline{X}^2$ is done, sample standard deviation is calculated using

$$Sample\ Standard\ Deviation\ =\ Standard\ Deviation\ *\ \sqrt{\frac{N}{N-1}}$$

$$Standard\ Deviation\ =\ \sqrt{(\overline{X^2}) - \overline{X}^2}$$

  And the value is written to the shared memory. This process reads mean from shared memory written by Process 4.

- Process 3 - This process creates a temporary copy of the array and sorts it, then it calculates the median and writes it to the shared memory.
- Process 4 - This process is created by Process 2 and calculates the mean and writes it to shared memory which is then read by Process 2 and Process 1(main).

th-stat.cpp

- Global variables include mean, median, standard deviation, number of values, and handles for threads calculating mean, median and standard deviation.
- The main thread starts the threads for mean, median and standard deviation.
- The mean-thread starts calculating the mean($\overline{X}$), the median thread copies the values and starts sorting, the std_dev thread starts calculating mean of square of values ($\overline{X^2}$).
- After calculating $\overline{X^2}$, std_dev (thread) waits for mean (thread) to terminate and then uses the same equations to calculate sample standard deviation.
- All the values are written in and read from global variables.

Analysis and Explanation

The time taken by both programs is nearly equal with multiprocessing taking a little extra time. The graph for CPU time and overall time is shown below (clp time was used for this).

The reason for this is because creating a new process is expensive. When a new process is created, everything has to be copied, including the code, global variables and heap. However, this is not the case with multithreading. In case of multithreading, the code, global variables and heap are shared and need not be copied. Thus saving some time at

the start of the execution. This time difference generally increases and multi-processing becomes more expensive with increase in the number of tasks.