## PROBLEM DESCRIPTION

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

Input text files was made by using rand function in separate c program with N values ranging from 1*10^6 to 5*10^6.

The primary difference is that threads within the same process run in a shared memory space, while processes run in separate memory spaces.Threads are not independent of one another like processes are, and as a result threads share with other threads their code section, data section, and OS resources (like open files and signals). But, like process, a thread has its own program counter (PC), register set, and stack space.

1.Process Program

- The program produces 1 parent and its 3child processes.
- Inter Process Communication mechanism is used in the program using POSIX shared memory
- t0 and t1 are pointers of datatype timeval .
- The child processes works in parallel to compute mean , median , standard deviation each.
- The parent process waits till the completion of all the children and displays the output(that is mean ,median and standard deviation.
- Mmap function which is a POSIX-compliant Unix system call maps the memory .This means that writes to a mapped area in

one process are immediately visible in all related processes hence mmap is used for inter process communication

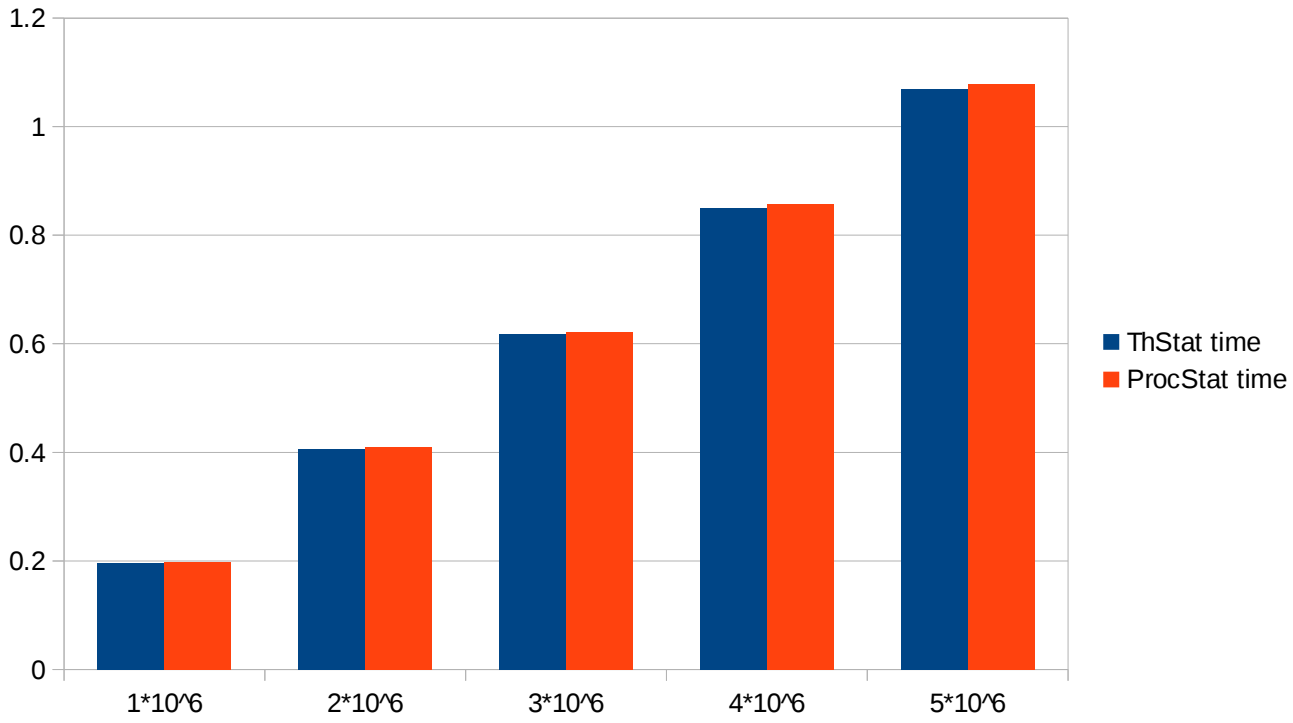
- gettimeofday() function records the current time stamp.
- The dynamically allocated array(data) takes input from a text file for further computations of mean ,median and standard deviation.
- Wait() function-This command can be useful where part of a script can execute in parallel to implement a barrier where an upcoming section depends on the successful completion of the preceding sections.

2.Threads Program

- Multithreading is the feature that allows your computer to run two or more programs concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution.
- **pthread_create** creates a new thread and makes it executable.
- **pthread_exit** is used to explicitly exit a thread. Typically, the pthread_exit() routine is called after a thread has completed its work and is no longer required to exist.
- The **pthread_join()** subroutine blocks the calling thread until the specified thread_id thread terminates. When a thread is created, one of its attributes defines whether it is joinable or detached. Only threads that are created as joinable can be joined. If a thread is created as detached, it can never be joined.
- The functions mean , median and std_dev are called by the threads created in the main function. The threads execute these functions in parallel and when all the threads complete their computation ,the output is stored in a text file.

## ANALYSIS OF OUTPUT

| N value | ThStat time | ProcStat time |
|---|---|---|
| 1*10^6 | 0.195508 | 0.198207 |
| 2*10^6 | 0.405711 | 0.410147 |
| 3*10^6 | 0.618279 | 0.621467 |
| 4*10^6 | 0.849348 | 0.856585 |
| 5*10^6 | 1.06923 | 1.0787 |




Case A- A program is divided into different processes.
Case B- A program is divided into different threads.

Case B takes less time to complete the execution rather than case A. It is because communication between multiple threads is easier, as the threads shares common address space while in process we have to follow some specific communication technique for communication between two process