CO21BTECH11002
Aayush Kumar
Assignment 1

SAM1:

Each thread gets a starting index and a vector to store the prime numbers. If there are ‘m’ threads then we initialize the starting indices as 1, 3, …, 2m-1, ignoring the even numbers. Now, each thread checks the numbers in a step size of 2m. So, after checking 1, 3, …, 2m-1, threads will check 2m+1, 2m+3, …., 4m-1 and so on.

For each number, the thread checks for primality in O(sqrt(n)). It uses the fact that if a number is not prime then it can be divided into two factors ‘a’ and ‘b’. Now, at least one of ‘a’ and ‘b’ must be smaller than or equal to sqrt(n). So, if we are not able to find a factor less than or equal to sqrt(n), the number must be prime.

In the main function, we read input from files, initialize and run the threads, and wait for them to finish executing and join them.

DAM1:

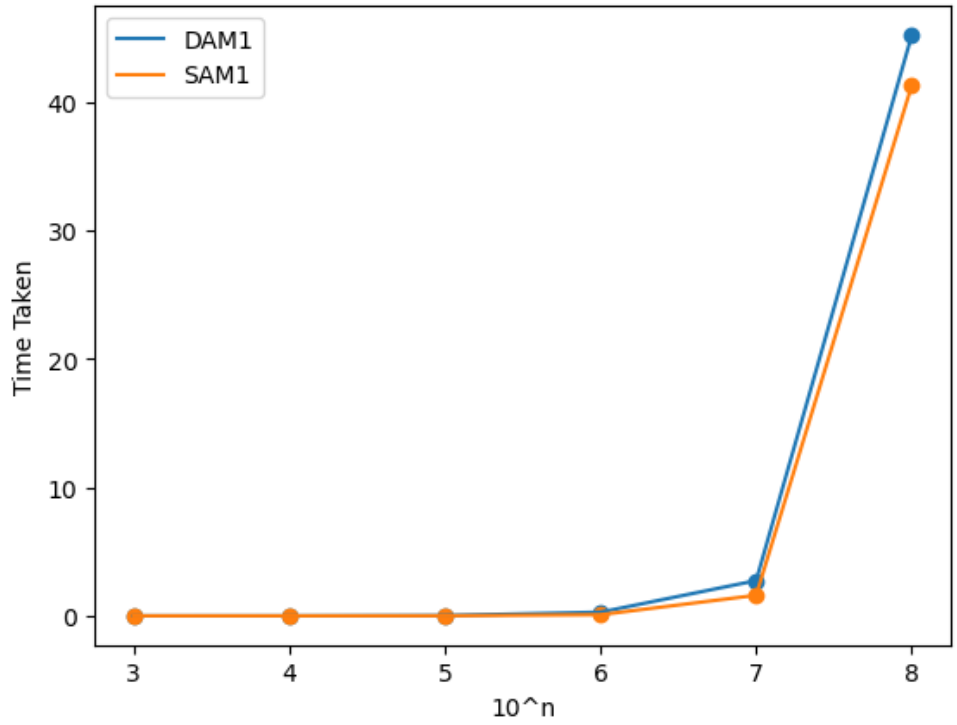
In this method, we use a semaphore ‘mtx’ to access the next number that is to be checked for primality. After acquiring the lock, the value of the number to be checked is increased by 2 to skip the even numbers. Each thread must acquire the lock before changing the value of the number to be checked to make sure that no two threads parallelly alter the value or get the same number to check.

Method to check if the number is prime or not is the same as in SAM1, in O(sqrt(n)).

In the main function, we read input from files, initialize and run the threads, and wait for them to finish executing and join them.

Results:

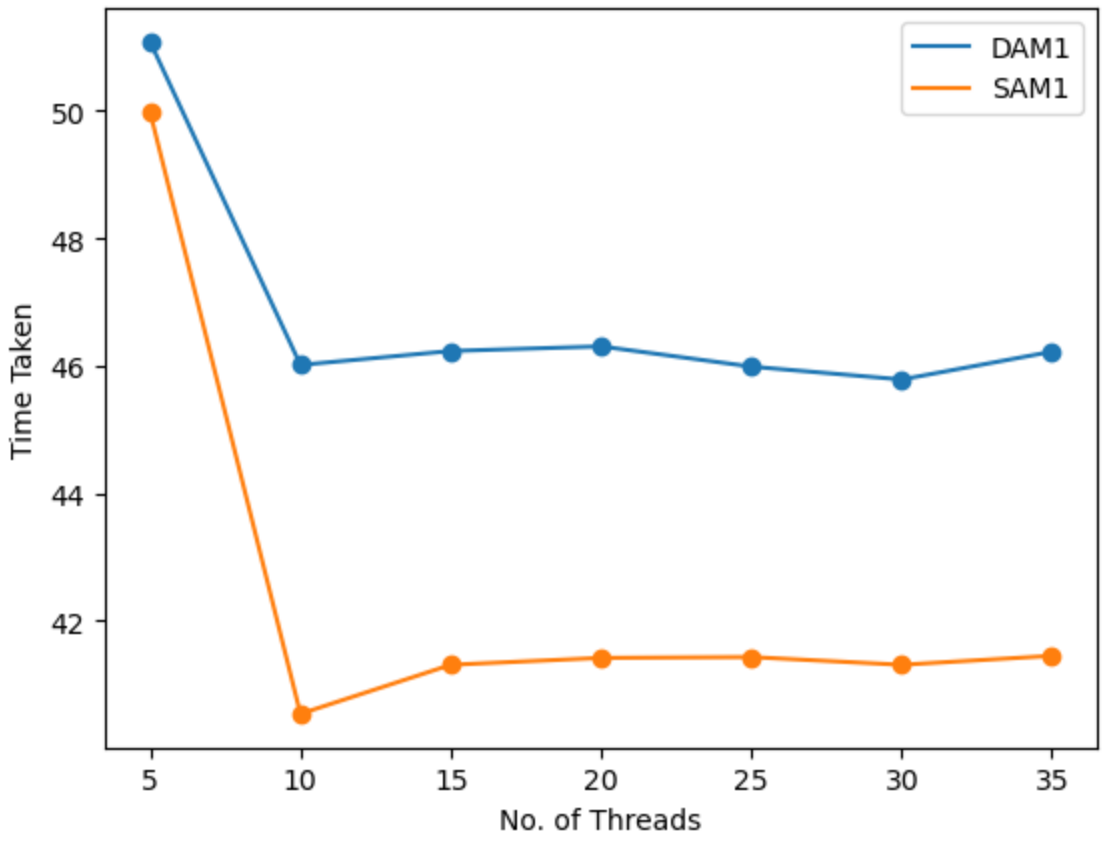
Time Taken (in seconds) vs the number of numbers to check keeping number of threads fixed at 10:



As we increase the number of numbers to be checked, the load on each thread increases, and hence, total time taken also increases.

SAM1 performs better than DAM1 since in DAM1, each thread has to wait to acquire the lock before it gets the number to check.

Time taken (in seconds) vs number of threads keeping numbers to be checked fixed at 1e8:



On increasing the number of threads, initially the time decreases since the load on each thread decreases. But, as we increase the threads further, the overhead for thread creation and context switching increases, hence, we do not see any further improvements in the time taken.
Also, DAM1 performs poorer than SAM1 since in DAM1 the threads have to wait for the lock to be available before they start checking any number.