

We will perform three tests, each consisting of multiple subtests. The tests will be performance across varying sized arrays with a constant number of processes or threads, constant sized arrays with variable number of processes or threads, and a test of extremes.

Test 1: variable array size, constant number of processes and threads

These tests will include 5 different sized intervals, 50, 100, 150, 200, and 250. They will be tested on the size interval * #procs/threads. This test will show that the search takes roughly the same amount of time regardless of the size of the interval and array if the number of processes or threads is consistent.

1. 5 Processes/Threads

- ./(proc/thread) 250 50
- ./(proc/thread) 500 100
- ./(proc/thread) 750 150
- ./(proc/thread) 1000 200
- ./(proc/thread) 1250 250

2. 10 Processes/Threads

- ./(proc/thread) 500 50
- ./(proc/thread) 1000 100
- ./(proc/thread) 1500 150
- ./(proc/thread) 2000 200
- ./(proc/thread) 2500 250

3. 15 Processes/Threads

- ./(proc/thread) 750 50
- ./(proc/thread) 1500 100
- ./(proc/thread) 2250 150
- ./(proc/thread) 3000 200
- ./(proc/thread) 3750 250

4. 20 Processes/Threads

- ./(proc/thread) 1000 50
- ./(proc/thread) 2000 100
- ./(proc/thread) 3000 150
- ./(proc/thread) 4000 200
- ./(proc/thread) 5000 250

Test 2: constant array size, variable number of processes and threads

To activate this test mode, input -1 as the interval. The program will calculate all intervals (up to 50 processes/threads maximum) that the array is divisible by and run search on them. EX) For an array of size 100, the possible intervals are 100, 50, 25, 20, 10, 5, 4, 2 which correspond to 1, 2, 4, 5, 10, 20, 25, 50 procs/threads respectively. (Graph will be runtime for the y-axis and number of processes for the x-axis). Run this on arrays of various sizes (each generating a graph) to see the overall trend of number of processes or threads and runtime.

1. Various sized arrays
 - `./(proc/thread) 100 -1`
 - `./(proc/thread) 250 -1`
 - `./(proc/thread) 500 -1`
 - `./(proc/thread) 1000 -1`
 - `./(proc/thread) 5000 -1`

Test 3: Extremes

This test will involve comparing the runtimes of processes and threads on larger arrays. The interval will remain constant at the maximum of 250, and the array size will increase. We expect that processes will get much worse faster than threads.

1. Extremes test
 - `./(proc/thread) 10000 250`
 - `./(proc/thread) 12500 250`
 - `./(proc/thread) 15000 250`
 - `./(proc/thread) 17500 250`
 - `./(proc/thread) 20000 250`