Zhe Yan (yanz9)
Haotian Zhan (zhanh)
Clara Peng (pengr)

Project-Analysis

1. (4 points) Of the four simulated algorithms, which algorithm is the "best" algorithm for CPU-bound processes? Which algorithm is best-suited for I/O-bound processes?

   RR is the "best" algorithm for CPU-bound processes. Every process gets an equal share of the CPU. Besides, there is no starvation since RR is cycling in nature.

   SRT is best-suited for I/O-bound processes. Processes with short CPU burst time have higher priority, which means I/O bursts are prioritized.

2. (4 points) For the SJF and SRT algorithms, what value of α produced the "best" results?

   Argument line: 8 2 0.01 256 4 alpha 128

```
----------alpha = 0.1----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 50.707 ms
-- average turnaround time: 138.691 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 36.152%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 50.418 ms
-- average turnaround time: 138.546 ms
-- total number of context switches: 315
-- total number of preemptions: 11
-- CPU utilization: 35.970%
----------alpha = 0.2----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 54.368 ms
-- average turnaround time: 142.352 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 35.691%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 55.020 ms
-- average turnaround time: 143.227 ms
-- total number of context switches: 321
-- total number of preemptions: 17
-- CPU utilization: 35.845%
```

Graph 1

```
----------alpha = 0.3----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 59.062 ms
-- average turnaround time: 147.046 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 35.804%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 48.859 ms
-- average turnaround time: 137.079 ms
-- total number of context switches: 322
-- total number of preemptions: 18
-- CPU utilization: 35.942%
----------alpha = 0.4----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 59.240 ms
-- average turnaround time: 147.224 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 35.790%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 49.329 ms
-- average turnaround time: 137.549 ms
-- total number of context switches: 322
-- total number of preemptions: 18
-- CPU utilization: 35.847%
----------alpha = 0.5----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 57.888 ms
-- average turnaround time: 145.872 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 35.829%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 57.076 ms
-- average turnaround time: 145.336 ms
-- total number of context switches: 325
-- total number of preemptions: 21
-- CPU utilization: 35.595%
```

Graph 2

Based on Graph1 and Graph2, both SJF and SRT have the shortest average wait time at alpha = 0.3. Therefore, alpha = 0.3 produces the “best” value.

Zhe Yan (yanz9)
Haotian Zhan (zhanh)
Clara Peng (pengr)

3. (4 points) For the SJF and SRT algorithms, how does changing from a non-preemptive algorithm to a preemptive algorithm impact your results?

```
----------2 processes----------
Algorithm SJF
-- average CPU burst time: 103.027 ms
-- average wait time: 12.973 ms
-- average turnaround time: 120.000 ms
-- total number of context switches: 37
-- total number of preemptions: 0
-- CPU utilization: 23.424%
Algorithm SRT
-- average CPU burst time: 103.027 ms
-- average wait time: 12.973 ms
-- average turnaround time: 120.000 ms
-- total number of context switches: 37
-- total number of preemptions: 0
-- CPU utilization: 23.424%
----------4 processes----------
Algorithm SJF
-- average CPU burst time: 94.885 ms
-- average wait time: 22.519 ms
-- average turnaround time: 121.404 ms
-- total number of context switches: 52
-- total number of preemptions: 0
-- CPU utilization: 30.200%
Algorithm SRT
-- average CPU burst time: 94.885 ms
-- average wait time: 21.981 ms
-- average turnaround time: 120.942 ms
-- total number of context switches: 53
-- total number of preemptions: 1
-- CPU utilization: 30.316%
----------6 processes----------
Algorithm SJF
-- average CPU burst time: 89.518 ms
-- average wait time: 39.922 ms
-- average turnaround time: 133.440 ms
-- total number of context switches: 141
-- total number of preemptions: 0
-- CPU utilization: 29.744%
Algorithm SRT
-- average CPU burst time: 89.518 ms
-- average wait time: 39.213 ms
-- average turnaround time: 133.099 ms
-- total number of context switches: 154
-- total number of preemptions: 13
-- CPU utilization: 30.095%
----------8 processes----------
Algorithm SJF
-- average CPU burst time: 83.984 ms
-- average wait time: 57.888 ms
-- average turnaround time: 145.872 ms
-- total number of context switches: 304
-- total number of preemptions: 0
-- CPU utilization: 35.829%
Algorithm SRT
-- average CPU burst time: 83.984 ms
-- average wait time: 57.076 ms
-- average turnaround time: 145.336 ms
-- total number of context switches: 325
-- total number of preemptions: 21
-- CPU utilization: 35.595%
----------10 processes----------
Algorithm SJF
-- average CPU burst time: 84.760 ms
-- average wait time: 77.312 ms
-- average turnaround time: 166.072 ms
-- total number of context switches: 391
-- total number of preemptions: 0
-- CPU utilization: 45.919%
Algorithm SRT
-- average CPU burst time: 84.760 ms
-- average wait time: 79.913 ms
-- average turnaround time: 169.092 ms
-- total number of context switches: 432
-- total number of preemptions: 41
-- CPU utilization: 46.130%
```

Graph 3

Based on the Graph 3 above, we can conclude that the average waiting time of fewer processes is shorter run by SRT. However, SJF may be a little faster than SRT processing more processes like 10 processes.

4. (6 points) Describe at least three limitations of your simulation, in particular how the project specifications could be expanded to better model a real-world operating system.

   Limitation 1: We only simulate CPU bursts. I/O bursts are not considered. There could also be some preemptions or something for the I/O bursts Queue.

   Limitation 2: Processes with different initial "tau" time are not tested since we can only test one alpha in one test. However, it will be more complex in the real world.

   Limitation 3: A seed is set for this random test, so the testing result can be considered as pseudo-randomly. A real-world operating system will be really randomized.

5. (6 points) Describe a priority scheduling algorithm of your own design (i.e., how could you calculate priority?). What are its advantages and disadvantages?

   We can create a new I/O burst First algorithm. In this algorithm, we need to sort the ready queue before the process started. Besides, when a new process arrived, we need to check its I/O burst time with the current running process before entering to the ready queue. If it has less I/O burst time than the I/O burst time of current running process, a preemption occurs. Then, the current running process is added back to ready queue.

Advantage: If there are not so many processes, waiting time and turnaround time can be greatly reduced.

Disadvantage: It is not so efficient if there are many processes or the CPU burst time is very long, which is the same as FCFS.