Alex Blair

Mckenna Galle

Julia Abbott

Operating Systems

Program Specs.

[Jabbott19@my.whitworth.edu](mailto:Jabbott19@my.whitworth.edu)

[Mgalle19@my.whitworth.edu](mailto:Mgalle19@my.whitworth.edu)

Scheduling Algorithm Simulation

Requirements:

* Multi-core processor
* Windows operating system
* Java JDK and JRE
* Useable input and output devices such as a keyboard and mouse.
* Main memory
* Disk memory
* Data structures such as (Vectors, Queues, Array Lists, Priority Queue)

Language:

Our group has chosen to use Java so that we can easily separate each algorithm into a class and

In this simulation we aim to simulate multiple different scheduling algorithms for different processes that a PC may have running. We will keep track of a number of statistics for these algorithms. These include, program ID, CPU burst time in milliseconds, IO burst in milliseconds, and CPU utilization, average wait times, average execution times, throughput, average turnaround time, and average response time. We will design our own method of creating a PCB or Program control block to control all running processes. We will create classes for each algorithm and use these objects in conjunction with the PCB to begin the simulation. If possible we would like to separate each algorithm into its own thread so that they can run concurrently and then compare results at the end. As far as the data, we will write the data to a file and this file will be our results for presentation.

Assumptions:

* For this project we will assume that a process has two states, ready and running.
* We will also assume that the processor is a multicore processor

PCB Structure:

* Variables include: PID (Process ID), State, Program counter, Memory info
* IO status

PCB Methods:

* Some methods include: Get state, get counter number, get Memory info, get IO status,

Each CPU register will be its own class and inherit from the parent PCB class.

The algorithms for scheduling will all be separate classes as well and inherit from the parent PCB class to have access to the PCB functionality.

Multiprocessor Approach:

For the multiprocessor approach we plan to implement a Load Sharing algorithm where we implement

Uniprocessor approach:

UML:

Our UML is in our GitHub Repository.