**CS 475 Operating Systems**

**CPU Scheduler Dispatch Simulation Project**

### NOTE: This is a rigorous project and requires extensive time for programming, testing and analysis. DO NOT wait until the last minute. It will require significant outside time to complete.

### Deliverables March 18: Part A

* System design documents

### Deliverables March 25:

* Walkthrough/Demonstration of your simulation model with informal discussion of your findings
* Group benchmarking report
* Individual assessment write-ups

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In this assignment, you will design and implement a ***discrete event model*** simulating CPU dispatch scheduling (short-term and dispatch management) for a multi-core system. You are to perform extensive test runs, collect data, perform benchmark analysis and report those findings. You’ll analyze the performance of various queue configurations (using different scheduling algorithms) and various time quantums, with the goal of finding the optimal configuration. Finally, you are to report your final system assessment as though you are a systems analyst.

* Work in **teams** of 2 or 3.
* You may use any **programming language of your choice**.

**Objectives:** Integrate OS theory and practice in the design of a substantial model/simulation

Gain a better understanding of the process scheduling and dispatch process.

Learn discrete event simulation concepts

Learn how to thoroughly analyze a system and present a meaningful assessment

Effective teamwork and project management

### Requirements:

**Part A: Model Design & Implementation**

You are to create a simulation program that models a CPU dispatch scheduling. This includes distribution to multiple CPUs and scheduling on each single CPU processor. Realize any model is a partial representation – you must make assumptions and cleverly design the most accurate representation possible. For instance, it is a **discrete** **event** model (see below in timing).

**PCB Structure**

Processes are ***model entities***, implemented as objects including the necessary accounting structures. For instance a PCB structure tracking the control and accounting information you deem necessary (i.e. PID, priority, state, CPU burst time, accumulated T, W, R, etc.) Note, it is not necessary to include all elements of an actual PCB – only the elements required for your model. You’ll also need a Process table that reference PCBs with PIDs.

**Data Files**

You’ll need to generate random data files of process information. Each line represents one process. The first element is a unique PID. Each line consists of that Process PID, Arrival time, a sequence of numbers representing CPU bursts and I/O bursts in milliseconds.

Example:

PID #, Arrival t, CPU t, IO t, CPU t, IO t, CPU t

PID #, Arrival t, CPU t

PID #, Arrival t, CPU t, IO t, CPU t, IO t

**Multiprocessor Distribution Assignment:**

Choose one of the 4 approaches described in Ch 10 to design and implement. You are to consider issues associated with multiprocessor processing.

NOTE: Remember this is a model. Though you are representing multicores, you need not multithread the application for each core. If you see an opportunity to improve your simulation model with multithreads, go for it (but it is not required).

**Uniprocessor multiprogramming dispatch scheduler**

Implement the necessary queues. Required is a multilevel feedback queue consisting of multiple queues. You must factor in some form of aging and priority feedback. You are to have multiple scheduling algorithms including as a minimum: multiple RR with different time quantums, FCFS. You could also include SPN or SRT.

**This is where you are to be creative, try different arrangements, test them and come up with your best solution.**

**Discrete Event Model:**

* + Implement time quantums as ***discrete events***, advanced by the designated block of time, ***NOT a continuous interval clock module***.
  + Also model context switches as ***discrete events*** – again as a designated block of time. Don’t try to model all the steps of a context switch.
  + Processing of CPU bursts are also treated as ***discrete events***.
  + You’ll need to shuffle processes in/out of queues upon IO bursts. Again, treat the IO burst times as **discrete events**.
  + You are to execute multiple runs, to collect a good representative set of process sampling. Execute each run until the given processes have completed, capturing all necessary time measurements for analysis and write the data to files for later benchmarking analysis.

**Assumptions:**

An important part of simulation modeling is to clarify assumptions made. You’ll likely have additional assumptions stated in the Team Benchmarking Report, however here are some example assumptions you may make.

1. A process can be either in the state “ready” or “waiting” for IO burst.
2. Upon an IO Operation processes are moved to a waiting queue. After IO completion, a process is transferred to a ready queue.

Again remember the bursts (CPU and IO) are treated as discrete events – therefore becomes

just a matter of moving processes back and forth (not waiting for actual time to pass).

**Execution and Collection of Data:**

Dynamic execution of processes – capture in a file per execution, for each process the metrics necessary for analysis.

At the end of simulation, the program should calculate and print such things as:

* 1. Throughput time
  2. Average turnaround times
  3. Average wait times
  4. Average response times
  5. Average context switch time
  6. Processor utilization - U [%]
  7. Speedup over multiple CPUs. NOTE: test varying number of CPUs

**Part B: Benchmarking Analysis & Report**

* Testing is an important part of Simulation. **DO NOT underestimate the time required for this portion**. Allow sufficient time to make **many sets of simulation runs** (remember you will be randomly generating process times and deriving averages). Experiment with:
  + Varied time quantums
  + Analysis of different scheduling algorithms, variations of process allocation to specific queues
* Collect the necessary data during runs to perform thorough evaluation.
* Speedup (over different numbers of CPUs)
* CPU Utilization

**Team Benchmarking Report** with numerical data collection, illustrated graphs and stated assumptions.

Make it look PROFESSIONAL!

* + As with any simulation model - You will need to make **assumptions** for your system such as timing issues, arrival policies, etc. State those clearly in this document.
  + Create **graphs** (i.e. Excel) that plot the required analysis of **average Turnaround (Tt), average Wait (Wt) and average Response (Rt )** given a **significant number of runs** with **different time quantums**. These will be plotted in 3 graphs with metrics (e.g. average Tt , Wt , Rt on they-axes, respectively) per time quantums on the x-axis.
  + **Average Throughput (Th)** for various algorithms.
  + **Speedup** analysis with differing CPU configurations.
  + … Other analysis you see pertinent
  + **Report data collection** for the graphs. You’ll want to collect data in files and then calculate averages.

**Part C: Individual write-ups and discussion of your findings**

A written discussion from your **individual perspective**. This is where you step into an Analyst role.

**a.)** Give a **summary** discussing your design, implementation methods, observations, findings, your conclusions.

**b.) Given the analytical findings, what were your observations?** For instance, how did you handle such things as queue distribution, switching, etc. What quantum resulted in the best average turnaround times, did it favor shorter versus longer jobs? What configuration resulted in the best response time, in the least wait time? What effect did varied quantum times have? What differences did you see with predominantly short burst time processes with more context switches versus predominantly longer burst time processes? Other important observations are what time quantum resulted in the best average turnaround times, did it favor shorter versus longer jobs? What effect did varied quantum times have? How did you handle the multi-core scheduling? Discuss optimal findings.

c**.) Systems Assessment and suggestions**. You are the **Systems Analyst.** After all of this, what is your personal assessment of the design and implementation given your findings? What are strengths/weaknesses of your system? What are potential improvements that should be made to the system? What is an optimal configuration?

**d.) What role did you play on the project?** What did you contribute? How effective did your team work?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Grading Rubric/Criteria**

**Part A: System Design Documents (20 points)**

20 points System Design and Project Management Plan

15 System requirements

Process entity support (Process Table, PCB, etc)

Description of multicore process distribution approach

Description of Uniprocessor multiprogramming scheduling

Diagram of multilevel queue configuration & scheduling algorithms

UML Diagrams

05 Project Management Plan

Deliverables each team member is responsible for?

Milestone deliverable dates

**Part B: Code Review (45 points)**

45 points

Respond carefully to the questions given for the code review

**Part C: Team Work and Project Management (10 points)**

10 points How well did you work as a team member

Demonstrated use of effective project management

**Part D: Group Benchmarking Analysis Report (15 points)**

15 points Graphs of T, W, and R;

Throughput for sample time frames

Statement of Assumptions

**Part E: Individual Writeup (10 points)**

10 points Observations and conclusive system assessment

Your suggestion for optimal configuration

Role and contributions to the project

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**100 Total Possible**