AI3163 Course Project:

AI-Assisted Job Scheduling in Operating Systems

1. Project Overview

Job scheduling is a key component of modern operating systems, impacting performance metrics like waiting time, turnaround time, throughput, and resource allocation fairness. In this project, you will simulate job scheduling algorithms in an OS environment and enhance the scheduling decision-making process using machine learning. You will:

1. Train a machine learning model to predict job runtimes based on synthetic features.
2. Implement classic OS scheduling algorithms: First-Come First-Served (FCFS) and Round-Robin (RR).
3. Implement a Predicted-Shortest-Job-Time (Predicted-SJF) scheduler that uses the runtime predictions from your trained model.
4. Simulate different workloads and evaluate the performance of your schedulers.

Note: You are encouraged to work individually on this project. If you choose to work in a group, each group may have ***no more than 2 students****, and you must complete the* ***additional required tasks****.*

2. Data Description

You are provided with two files (training_jobs.csv and testing_jobs.csv). You should use training_jobs.csv to train your predictor and testing_jobs.csv to evaluate the performance of your schedulers. Each record in the file is explained as follows.

Attribute	Meaning
job_id	**Unique identifier for the job**
arrive_time	Time when job arrives in the system
model_size	Synthetic feature related to job complexity
batch_size	Synthetic feature related to job batch processing
dataset_size	Synthetic feature indicating the dataset size being processed
epochs	Synthetic feature related to training epochs
uses_gpu	Indicates whether the job uses GPU (1 = yes, 0 = no)
true_runtime	Actual job runtime, which your model will predict

3. Task description

(1) Load and sort jobs

For each job csv file:

a) For each job file, load every row into a job object.

b) Sort the jobs by their *arrive_time* to simulate the order in which jobs arrive at the system.

(2) Implement FCFS and RR (Baselines)

a) **FCFS**: Implement this non-preemptive scheduling algorithm, where jobs are executed in the order of their arrival.

b) **RR**: Implement the Round-Robin scheduling algorithm with a time quantum that you define.

(3) Build your Predictor

a) Train a machine learning model to predict the true_runtime of jobs based on the provided features in training_jobs.csv.

b) Implement a function *predict_runtime(job)* that returns the predicted runtime of a job.

c) You can use any machine learning model (e.g., decision trees, regression models) but should explain your choice of algorithm.

(4) Implement the Predicted-SJF Scheduler (Non-preemptive)

a) Since jobs arrive at different times, the scheduler should dynamically select the job with the shortest predicted runtime from the set of **jobs that have already arrived**.

b) You must implement this as a **non-preemptive scheduler**, meaning that once a job starts running, it continues until completion. Job selection should be based solely on the runtime predictions produced by your model.

(4-2) Implement the Predicted-Shortest-Remaining-Time-First Scheduler (preemptive)

a) This is an additional requirement only for students who complete the project in a group.

b) As jobs arrive at different times, the scheduler should always select the job with the shortest predicted remaining runtime among all currently arrived jobs. If a new job arrives with a shorter predicted remaining time, the scheduler should preempt the currently running job and switch to the new one.

(5) Compute Scheduling Metrics

For each scheduling algorithm, compute and display the average waiting time and average turnaround time.

4. Files to submit

1. Implementation Files **(with comments)**:

(1) Predictor (train + predict)

(2) FCFS Algorithm

(3) RR Algorithm

(4) Predicted-SJF Algorithm

(5) Metric Computation

(6) Predicted-Shortest-Remaining-Time-First Algorithm (*required only for students working in a group*)

2. Your report must include the following parts:

(1) Explain how you load and store the job data.

(2) Explain the implementation of FCFS and RR algorithms.

(3) Explain your predictor:

- What machine learning model did you use?
- Why did you choose that model?
- Any preprocessing or feature handling you performed.

(4) Explain how the Predicted-SJF scheduler works. Discuss how the scheduler handles jobs that arrive at different times and how the job with the shortest predicted runtime is selected.

(4-2) (*Required only for students working in a group*) Explain how this preemptive scheduler works and how it selects the job with the shortest predicted remaining runtime.

(5) Include a screenshot of the printed average waiting time and average turnaround time for each scheduler.

Compare and discuss the differences in performance.

(6) The report must be a PDF file named as: <student id>_<name>.pdf. For example, 3110101314_Jacky_Chan.pdf. If you complete the project in group, the report file should be name as <student id 1>_<name 1>_<student id 2>_<name 2>.pdf.

***For students working in groups, you are required to:**

a) Completed the additional task about the Predicted-Shortest-Remaining-Time-First scheduler.

b) Declare your role in the project and what you contributed.

c) Discuss the relationship between AI and operating systems, focusing on how AI can enhance OS performance. Consider potential research directions for AI's integration into computer architectures and operating systems. **Each student should contribute to a section of this discussion.**

d) Submit only one set of files to iSpace (submitted by one group member).

*Note:

(1) **Plagiarism will not be tolerated**. If you are caught copying someone else's code, you will receive a grade of 0 for this project.

(2) For group projects, each student must contribute equally to the implementation and report.