Assignment 2: Design and Implementation of a New Scheduling Algorithm

1. Report (7 marks)

1.1 Introduction .... [/1]

1.2 Problem definition .... [/1]

1.3 Algorithm description including an example scheduling scenario ...... [/2]

1.4 Implementation .... [/1]

1.5 Evaluation .... [/1]

1.6 Presentation .... [/1]

2. Implementation (15 marks; ***NO MARKS***[1])

2.1 Client-server connection and handshaking successful .... [/1]

2.2 All test configurations are properly handled (scheduling of all jobs) .. [/2]

2.3 Scheduling algorithm's averages w.r.t. all three performance metrics (avg turnaround time, avg resource utilisation, total rental cost) outperform one or more baseline algorithms (i.e., values in yellow or green) .... [/2]

2.4* Performance with respect to the average turnaround time .... [/10]

** No marks will be given for 2.4 if 2.3 is not passed (i.e., if 2.3 = 0, then 2.4 = 0).*

*** A schedule is superior if the result is green. Marginally better results may be subject to manual inspection.*

3. Design at Code Level (6 marks)

3.1 Elegance (no redundant code, use of appropriate data structures) .... [/2]

3.2 Efficiency (efficient memory use, no magic numbers) .... [/2]

3.3 Readability (good naming, proper indentation, comments) .... [/2]

4. Project Management/Compliance (2 marks)

- Regular and genuine commits, Use of LaTeX template, page limit, formatting, README for extra steps, etc. .... [/2]

Total .... [/30]

[1] No marks will be awarded if the code doesn't compile or your client doesn't complete execution within 10 seconds for configuration files of up to 2000 jobs when tested on university servers, with execution time scaling linearly with input size (e.g., 15 seconds and 20 seconds for 3000 jobs and 4000 jobs, respectively.) You should remove print statements used for debugging purposes before the final submission to avoid slowdown.