Challenge #2
Job dispatching and scheduling

Problem outline

- Consider a computational cluster made up of
  1. One job dispatcher device;
  2. N servers, each equipped with its own processing and memory resources.
- The workload offered to the computational cluster is described by a dataset, taken from measurements on a production data center of Google and made publicly available.
- Here is the link where the dataset can be downloaded from: `https://github.com/MertYILDIZ19/Google_cluster_usage_traces_v3_Cell_a`
- The purpose of the challenge is to define a *dispatching algorithm* and *scheduling algorithms*, one for each server, so as to get the best possible *mean job response time*.

Workload dataset

The dataset extracted from Google public data is a five-column table in csv format, made up of 2329133 rows (please check).

Here is the content of each column.

1. JID: integer number representing the identifier of a job.
2. TID: an integer number between 0 and n_j, representing the identifier of tasks belonging to job j.
3. t_a: arrival time of task measured in milliseconds.
4. C: running time in seconds required to run the task on a Google Normalized Computing Unit (GNCU).
5. M: amount of memory required to run the task, expressed in Google Normalized Memory Unit (GNMU).

Assumptions (1/2)

- Each server has the same processing power μ, expressed in GNCU. Hence, the service time of a task is $X = C/\mu$.
- Each server has a memory amount of 1 GNMU.
- Pre-emption is allowed as well as server sharing, if you deem that any of these can be useful.
- It is not allowed to kill a job or a task, i.e., all tasks must be worked out eventually.
- You can assume that the dispatcher and servers know the exact service time of a task, upon its arrival.
- Delay and loss are negligible for message exchange among the dispatcher and the servers.

Assumptions (2/2)

- Each running task is assigned the memory space it requires as long as it is running. Swapping a task from running to standby and back requires negligible time.
- At any given time, the sum of all assigned memory workspaces to running tasks on a given server shall not exceed the overall memory of that server.

Metrics

- **Job response time** R: time elapsing since the arrival of the first arriving task of a job until all tasks belonging to that job have been fully served. The mean job response time $\overline{R}$ is obtained by averaging response times of all jobs.
- **Job slowdown** S: ratio of response time of the job to the sum of service times of all tasks belonging to the job. The mean job slowdown $\overline{S}$ is obtained by averaging slowdown values of all jobs.
- **Utilization coefficient of server** k, ρ_k: fraction of time that the server k is busy serving tasks. The overall mean utilization coefficient is $\rho = (\rho_1 + \cdots + \rho_N)/N$.
- **Messaging load** L: number of messages exchanged between the dispatcher and servers for a given task dispatching. The mean message load $\overline{L}$ is obtained by averaging message load values of all tasks.

Assignment

1. Set $N = 64$ and $\mu = 0.1$.
2. Algorithms.
   - Give a formal description of your dispatching algorithm.
   - Give a formal description of your server scheduling algorithms.
   - Provide motivation and rationale of your algorithms (i.e., why you think they should be good).
3. Performance evaluation
   - Give the values of $\overline{R}$, $\overline{S}$, ρ, $\overline{L}$, and a table with values of ρ_k, $k = 1, \dots, 64$.
   - Plot the empirical Complementary Cumulative Distribution Function (eCCDF) of R and S.
   - Repeat the above performance evaluation for: (i) your algorithms; (ii) baseline algorithms (for comparison purposes), namely, LWL dispatching and FCFS scheduling in all servers.

Hints

- Performance evaluation can be done by scanning orderly (according to arrival time) all tasks and applying the dispatching and scheduling algorithm.
- At each arrival, the status of the system must be updated and you must keep track of quantities needed to evaluate the required metrics
- The status of a server is given by
  - The list of tasks assigned to the server at time t.
  - The amount of work yet to be completed for each task.
  - The amount of memory of the server taken up by running tasks.
  - Unfinished work immediately after the arrival of task n, U_n

    $$U_n = \max\{0, U_{n-1} - T_n\} + X_n, \qquad n \geq 1,$$

    where $U_0 = 0$, T_n is the inter-arrival time between arrival $n-1$ and n, X_n is the service time or the n-th arriving task.

Delivery of the assignment

The delivery of the assignment consists of a **max 4 pages written report** in the form of a pdf file (Font size: 12 pt).

1. **PAGE 1** – Cover Page: nickname of your group, your given and family names, and enrollment numbers.
2. **PAGE 2-3** – Formal description of algorithm and their motivations[6].
3. **PAGE 4** – Performance metric values and plots, plus some concise comments.

Accuracy, conciseness, readability and quality of algorithms, written texts, plots will be fundamental in the evaluation.

[6] It is expected that you "explore" the data and get insight into what are key factors affecting the considered performance metrics. The insight you get from data exploration should guide you to good and rationally motivated policies.