Assignments

General Introduction

This assignment aims to enhance your understanding and practical application of Discrete event simulation and reinforcement learning. Through this task, you will gain hands-on experience in modeling and simulating complex systems, as well as implementing reinforcement learning algorithms to optimize decision-making processes.

It is mandatory to attend the oral exam with **at least one** completed project concerning the development and integration of a reinforcement learning agent into a discrete-event simulation of an industrial environment. Students may present more than one project if desired.

Students may choose to tackle one of the following problems or propose their own. In the latter case, you must consult the professor ahead of the exam to verify the validity of your problem (it may be too simple or too hard).

Students can complete the assignment individually or as part of a team. The oral exam will be conducted individually.

Key Objectives:

1. Develop proficiency in designing and executing discrete event simulations.
2. Understand the principles and applications of reinforcement learning.
3. Apply theoretical knowledge to real-world scenarios.
4. Improve problem-solving and analytical skills.
5. Collaborate effectively in a team setting.

Please note that the objective of the assignment is not necessarily to show a successfully trained RL agent, although it may be more interesting for both parties. Due to time and computational budget constraints, students are allowed to present an unsuccessfully trained agent. However, the work must demonstrate the effort put into solving the assignment.

Expected effort:

- Successfully implement a discrete event simulation of an operation management or supply chain environment.
- Adhere to the guidelines discussed in lectures when implementing the simulation environment.
- Conduct a thorough performance study of the system using methods discussed in lectures.
- Implement the reinforcement learning agent from scratch or use a library's black-box implementation.

- If using a library's black-box implementation, ensure you thoroughly understand the algorithm being used.
- You may use any extension of the RL algorithms discussed in the lectures, provided you understand the type of algorithm and its peculiarities.
- Experiment with different RL algorithms and different hyper-parameter combinations. You are not required to show a successfully trained agent but must demonstrate your effort in attempting to solve the assignment.

Inventory System

The first assignment focuses on the inventory system covered in the lectures. At least one reinforcement learning agent must be used to manage the replenishment policies of the warehouse.

The warehouse handles two different products, each sourced from a different supplier.

At the beginning of each day, the warehouse management must decide whether to place a replenishment order for each product. If an order is placed, the quantity to be ordered (in discrete units) must also be determined.

The objective is to minimize the overall cost.

Students must compare the performance of the RL-based solution with the performance of the s-min, S-max policy, for each product.

The simulation parameters are as follows:

- Demand inter-arrival times: exponential random variable, lambda 0.1
- K = 10
- i = 3
- h = 1
- π = 7
- First product demand distribution:
  - $D = \begin{cases} 1 & w.p. \ 1/6 \\ 2 & w.p \ 1/3 \\ 3 & w.p. \ 1/3 \\ 4 & w.p. \ 1/6 \end{cases}$
  - Lead Time: U(0.5;1)
- Second product demand distribution:
  - $D = \begin{cases} 5 & w.p. \ 1/8 \\ 4 & w.p \ 1/2 \\ 3 & w.p. \ 1/4 \\ 2 & w.p. \ 1/8 \end{cases}$
  - Lead Time: U(0.2;0.7)

Production System

The second assignment focuses on a production system similar to those covered in the lectures. The production system consists of six machines, each considered unbreakable (no maintenance required).

This production system is responsible for producing three different families of products. The product family determines the shop floor routing and the processing times for each machine.

First, students must implement a simulation of the environment managed using the "PUSH" policy: whenever a new customer order is received, it is immediately released onto the shop floor. The PUSH policy serves as the benchmark for comparison with the RL-based solution.

Secondly, students must implement an alternative environment where, upon receiving a customer order, the order is placed into a "pre-shop pool" (PSP). At regular intervals, a reinforcement learning (RL) agent will decide whether to release the most urgent order from the PSP into the shopfloor.

The objective is to achieve the same throughput as the PUSH system, while minimizing the WIP and with a comparable job tardiness and earliness.

The simulation parameters are as follows:

- 6 machines
- 3 product families
- Job arrival rate (exponential): lambda 0.65
- Families' weights:
    - F1: 10%
    - F2: 52%
    - F3: 38%
- Routings (probabilistic):

	WC1	**WC2**	**WC3**	**WC4**	**WC5**	**WC6**
F1	1	1	0	1	1	1
F2	0.8	0.8	1	0.8	0.8	0.75
F3	0	0	1	0	0	0.75

- Processing Times:
    - F1: gamma distribution, alpha=2, beta=2
    - F2: gamma distribution, alpha=4, beta=0.5
    - F2: gamma distribution, alpha=6, beta=1/6
- Due Dates:
    - Uniform: U(30, 50)