Manufacturing Line Simulation

Muhammed Kamel

20201701074

This report gives an overview of the simulation, a manufacturing line simulation done using the SimPy library in Python. The simulation represents a process of manufacturing with a set of stages: loading, machining, assembling, inspecting, and packaging. The goal is to simulate the flow of parts through these stages, accounting for potential machine breakdowns and repairs.

**Code Overview**

Libraries and Constants

The simulation makes use of the following libraries:

simpy: For event-based process simulation.

random: For generating random numbers to simulate breakdowns.

pandas: For storing and analyzing the simulation data.

Key constants used in the simulation include processing times for each stage, maintenance time, breakdown rate, number of workers, shift length, simulation time, and the number of product types.

PROCESSING\_TIMES = {

'loading': 5,

'machining': 10,

'assembling': 8,

'inspecting': 6,

'packaging': 4

}

MAINTENANCE\_TIME = 3

BREAKDOWN\_RATE = 0.1

NUM\_WORKERS = {

'loading': 2,

'machining': 3,

'assembling': 4,

'inspecting': 2,

'packaging': 3

}

SHIFT\_LENGTH = 8

SIMULATION\_TIME = 100

NUM\_PRODUCTS = 2

These define the number of workers available at each stage, the length of a work shift, the total simulation time, and the number of different product types being processed.

**Manufacturing Line Class**

The ManufacturingLine class models the entire manufacturing process. It has the following methods:

\_\_init\_\_(self, env): This will initialize the manufacturing line. It will prepare resources for each stage and for a repair team.

process\_part(self, part, product\_type): It will simulate how a part should be progressed to go through all stages.

repair\_machine(self, stage): It will simulate the repair process for a machine.

**Part Manufacturer and Setup Functions**

part\_manufacturer(env, line, part\_id, product\_type): It starts the process of manufacturing a part.

setup(env, num\_parts\_per\_type): It sets the simulation environment and will start the manufacturing processes.

env = simpy.Environment()

process = env.process(setup(env, 5))

env.run()

df = process.value

print(df)

The ManufacturingLine class uses simpy.Resource to initialize resources for each stage of the process. It also sets up a repair team that can handle two breakdowns simultaneously.

The process\_part method is in charge of handling a part as it passes through the stages. For each stage:

A request is made for the resource of the stage, indicating the number of workers.

The part is worked on for a time associated with the particular part type.

If a breakdown occurs (with a random determination), the repair process is called.

The repair\_machine method is in charge of repairs to the machine. It achieves that by making a request for the repair team resource. Meanwhile, it passes the time given as the maintenance time.

**Part Manufacturer and Setup**

The part\_manufacturer function initiates the processing of a part. The setup function initializes the ManufacturingLine and starts the manufacturing processes for the specified number of parts per product type.

**Running the Simulation**

This creates the environment, and sets up the process is initiated. The simulation runs for the given duration, and the results are dumped into a DataFrame, which is printed out.

This simulates, in detail, the manufacturing process: staging, probable breakdown, and repair. By using SimPy, the event-based simulation is made extremely efficient, and now with the help of the pandas DataFrame, the results are analyzed. This can be extended and modified to run for different manufacturing instances to optimize the flow.