

Q1:

A: a physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, or process

B: a sum total of all the living and non-living elements and their effects that influence human life

C: A system is a way of working, organizing, or doing something which follows a fixed plan or set of rules

D: a physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, or process

E: the use of a computer to represent the dynamic responses of one system by the behaviour of another system modeled after it.

F: approach to data centre management that supplements infrastructure management (DCIM) tools with engineering simulation tools

G: Discrete event simulation (DES) is a method of simulating the behaviour and performance of a real-life process, facility or system

H: metric that quantifies the proportion of correct predictions made by the model out of all predictions

I: a measurement of how accurate predictions or classifications a model makes on new, unseen data

J: attribute manifested by a sufficiently random process, and a sufficiently large sample size

Q2:

a. Determine the mean and variance of X.

The mean (expected value) of a random variable is given by:

$$E(X) = \sum x_i \cdot P(X=x_i)$$

And the variance is given by:

$$2] Var(X) = E[(X-\mu)^2]$$

where μ is the mean.

For the given random variable $0 \cdot 0.4 + 1 \cdot 0.6 = 0.6$ $E(X) = 0 \cdot 0.4 + 1 \cdot 0.6 = 0.6$

$$0 - 0.6)^2 \cdot 0.4 + (1 - 0.6)^2 \cdot 0.6 = 0.24$$
$$Var(X) = (0 - 0.6)^2 \cdot 0.4 + (1 - 0.6)^2 \cdot 0.6 = 0.24$$

b. Plot the probability density function (pdf).

import matplotlib.pyplot as plt

values = [0, 1]

probabilities = [0.4, 0.6]

plt.bar(values, probabilities, align='center', alpha=0.7)

plt.xlabel('X')

plt.ylabel('Probability')

plt.title('Probability Density Function (pdf) of X')

plt.show()

c. Plot the probability distribution function (CDF) of X.

import numpy as np

cumulative_probabilities = np.cumsum(probabilities)

plt.step(values, cumulative_probabilities, where='post', color='b')

plt.xlabel('X')

plt.ylabel('Probability')

plt.title('Cumulative Distribution Function (CDF) of X')

plt.show()

Q3:

1. Start:
 - Begin the simulation.
2. Initialization:
 - Set initial conditions, parameters, and variables.
 - Initialize simulation clock.
3. Event List:
 - Create an event list to manage the order of events.
4. Generate Initial Events:
 - Generate initial events to kickstart the simulation.
5. Simulation Loop:
 - While there are events in the event list:
 - Get the next event from the list.
 - Update the simulation clock to the time of the event.
6. Event Handling:
 - Determine the type of event (arrival, departure, etc.).
 - Execute the corresponding event procedure.
 - This may involve updating state variables, scheduling new events, or performing other actions.
7. Data Collection:
 - Collect relevant data for analysis.
 - This may include system performance metrics, statistics, or other output.
8. Termination Criteria:
 - Check if termination conditions are met.
 - This could be a specific simulation time, a number of events, or reaching a desired state.
9. End:
 - Finish the simulation.
10. Analysis and Reporting:
 - Analyze the collected data.
 - Generate reports or visualizations.
11. End of Flowchart:
 - End the flowchart.

Q4:

```
import simpy
import random
```

```
class SingleServerQueue:
```

```
    def __init__(self, env, arrival_rate, service_rate):
        self.env = env
        self.server = simpy.Resource(env, capacity=1)
        self.arrival_rate, self.service_rate = arrival_rate, service_rate
```

```
    def arrival_process(self):
        customer_id = 1
        while True:
            yield self.env.timeout(random.expovariate(self.arrival_rate))
```

```

        self.env.process(self.service_process(customer_id))
        customer_id += 1

def service_process(self, customer_id):
    with self.server.request() as request:
        yield request
        service_time = random.expovariate(self.service_rate)
        yield self.env.timeout(service_time)
        print(f"Customer {customer_id} served in {service_time:.2f} units at {self.env.now:.2f} units.")

def run_simulation(arrival_rate, service_rate, simulation_time):
    env = simpy.Environment()
    queue = SingleServerQueue(env, arrival_rate, service_rate)
    env.process(queue.arrival_process())
    env.run(until=simulation_time)

if __name__ == "__main__":
    run_simulation(arrival_rate=0.5, service_rate=0.7, simulation_time=10)

```

Q5:

1. Define the Road Network:

- Identify the main roads and intersections around KSU.
- Collect data on road lengths, lanes, speed limits, and road connectivity.

2. Traffic Flow Model:

- Choose an appropriate traffic flow model (e.g., microsimulation or mesosimulation).
- Consider traffic demand patterns, such as peak hours and weekdays vs. weekends.

3. Driver Behavior:

- Model driver behavior, including acceleration, deceleration, lane changing, and reaction to traffic signals.
- Incorporate factors like driver aggressiveness, patience, and adherence to traffic rules.

4. Traffic Signals and Controls:

- Implement traffic signal timings and controls at intersections.
- Consider adaptive signal control systems that can adjust timings based on real-time traffic conditions.

5. Data Collection:

- Gather real-world data on traffic patterns, including vehicle counts, speeds, and congestion levels.
- Use GIS data for accurate road geometry and topology.

6. Simulation Software:

- Choose a suitable simulation tool, such as VISSIM, AIMSUN, or any other traffic simulation software.
- Input the road network, traffic flow model, and driver behavior parameters into the simulation tool.

7. Validation and Calibration:

- Validate the simulation model against real-world data.
- Calibrate the model to ensure that it accurately represents observed traffic conditions.

8. Performance Metrics:

- Define performance metrics such as travel time, congestion levels, and queue lengths.
- Evaluate the impact of different scenarios, such as road closures or changes in traffic signal timings.

9. Accuracy Concerns:

- The accuracy of the simulation depends on the quality of input data and the realism of the chosen models.

- Calibration is crucial to ensure that the simulation results align with observed traffic conditions.
- The accuracy of driver behavior models can significantly impact the overall simulation accuracy.

10. Scenario Analysis:

- Conduct scenario analyses to assess the impact of different interventions (e.g., road expansions, signal changes).
- Consider future scenarios, such as increased traffic due to urban development.

11. Sensitivity Analysis:

- Perform sensitivity analysis to understand how changes in model parameters affect simulation outcomes.
- Identify critical parameters that significantly influence results.

12. Continuous Improvement:

- Update the simulation model regularly based on new data and changes in the road network or traffic patterns.
- Continuously refine and improve the model to enhance accuracy.

13. Engage Stakeholders:

- Involve local authorities, traffic management agencies, and other stakeholders in the simulation process.
- Seek feedback and validation from experts in traffic engineering and urban planning.

Q6:

Arrival and Service Times:

Customer	Arrival Time	Service Time
1	1	2
2	4	5
3	8	15
4	17	2

Calculate Finish Times and System State:

Time (t)	Event	Customer in Service	Queue State
1	Arrival (C1)	1	(empty)
3	Departure (C1)	2	(empty)
4	Arrival (C2)	2	(empty)
8	Arrival (C3)	2	3
10	Departure (C2)	3	(empty)
12	Departure (C3)	4	(empty)
17	Arrival (C4)	4	(empty)
19	Departure (C4)	-	(empty)

Performance Metrics:

1. System Throughput (X):
 - The number of customers served during the observation period.
 - $x=4X=4$ customers.
2. Total Busy Time (B):
 - The total time the server is busy.
 - $B=2+5+15+2=24B=2+5+15+2=24$ seconds.
3. Mean Service Time (T_s):

- Average time a customer spends in service.
 - $T_s = 244 = 6T_s = XB = 424 = 6$ seconds.
- Utilization (U):
 - Fraction of time the server is busy.
 - $U = B/20 = 2420 = 1.2U = 20B = 2024 = 1.2$.
 - Mean System Time (W):
 - Average time a customer spends in the system (waiting + service).
 - $W = \text{Total time spent waiting} = 24 + (3+6+7+0)4 = 10W = XB + \text{Total time spent waiting} = 424 + (3+6+7+0) = 10$ seconds.
 - Mean Number in the System (L):
 - Average number of customers in the system.
 - $L = 4 \cdot 10 = 40L = X \cdot W = 4 \cdot 10 = 40$ customer-seconds.

Q7:

- Collect Data:
 - Gather historical population data for Riyadh over the past 5 years.
- Calculate Birth and Death Rates:
 - Use the collected data to estimate annual birth and death rates.
- Estimate Immigration and Emigration Rates:
 - If available, collect data on immigration and emigration rates for Riyadh.
- Initialize Model:
 - Set the initial population (P_0) using the most recent population data.
- Apply Population Growth Equation:
 - Use the population growth equation to model population changes over the next 5 years.
- Compare with Real Data:
 - Compare the model's predictions with the actual population data for the corresponding years.

Comparison and Error Calculation:

Compare the model's predicted population with the actual population for each year.

Calculate the error as the absolute difference between the predicted and actual populations.

$\text{Error} = |\text{Actual Population} - \text{Predicted Population}|$

Q8:

- Initial Population (P_0): 7 million (based on the latest available data)
- Birth Rate (B): 2%
- Death Rate (D): 0.5%
- Immigration Rate (I): 1%
- Emigration Rate (E): 0.8%

Q9:

1. Discretize the Pipe:
 - Divide the pipe into a grid of nodes along the axial and radial directions.
2. Define Governing Equations:
 - Use the Navier-Stokes equations for incompressible flow (assuming steady-state, no-slip conditions).
 - These equations include terms for velocity, pressure, and viscosity.
3. Apply Finite Difference Approximations:
 - Discretize the spatial derivatives in the governing equations using finite difference approximations.
 - Common schemes include central differences for second-order derivatives.
4. Boundary Conditions:
 - Apply boundary conditions, considering the given inlet pressure and flow rate.
 - Implement no-slip conditions at the pipe walls.
5. Iterative Solution:
 - Set up an iterative solver to solve the system of discretized equations.
 - Common solvers include the Gauss-Seidel method or the Conjugate Gradient method.
6. Convergence Criteria:
 - Define a convergence criterion to determine when the solution has reached a steady-state.
7. Post-Processing:
 - Extract relevant information such as velocity profiles, pressure distribution, and flow patterns.

Abdulaziz Alhussaini

