Discrete Event Simulation Approach

CS1538: Introduction to Simulations

Simple Simulation: Single Server Queue

- Scenario: A fast-food restaurant with just a single server/cook and a single stove.
  - Customers arrive and wait in line for their turn.
  - The customer orders, then wait for the food to be prepared (service time may vary).
  - Customer picks up the food and leaves.

The Single Server Restaurant

- A few simplifying assumptions (for now)
  - Ignore: customer's party size, order complexity, money spent
  - Customers will patiently wait for their turns indefinitely
  - Customer arrival is independent and identically distributed (iid)
  - Service time is also iid.

- Some possible simulation objectives:
  - How often is the server idle?
  - What's the chance that a customer has to wait?
  - On average, how long is a customer's wait time?
    - How long is a customer's wait time *given that there already is a line*?

Modeling the Single Server Restaurant

- Need to model:
  - Randomized customer arrivals
  - Waiting in line
  - Randomized customer service times
- How to generate a random event?
- How to simulate the passage of time?

Time Advance Mechanisms

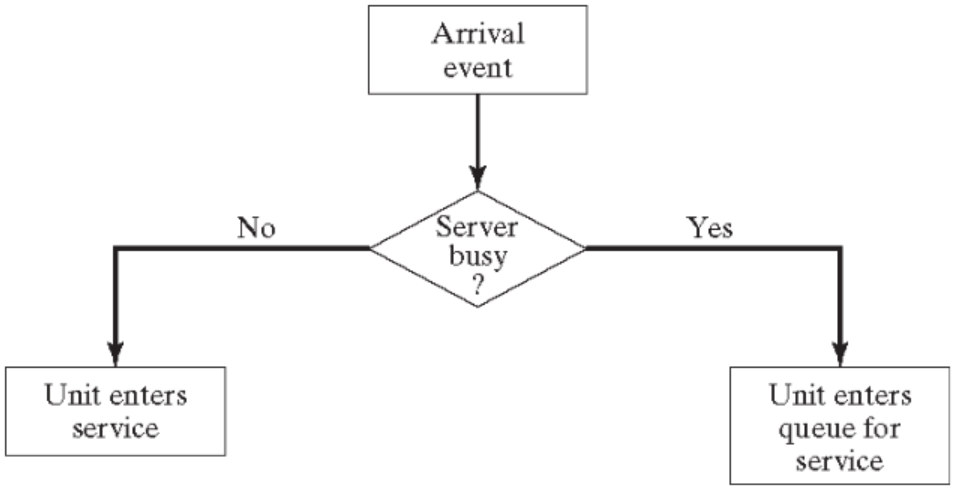
- Next-event time advance
  - Simulation clock initialized to zero
  - Determine the times of occurrences of future events
  - Advance clock to the most imminent of the future event
  - Update system variables
  - Update knowledge of the times for future events

Future Events List (FEL)

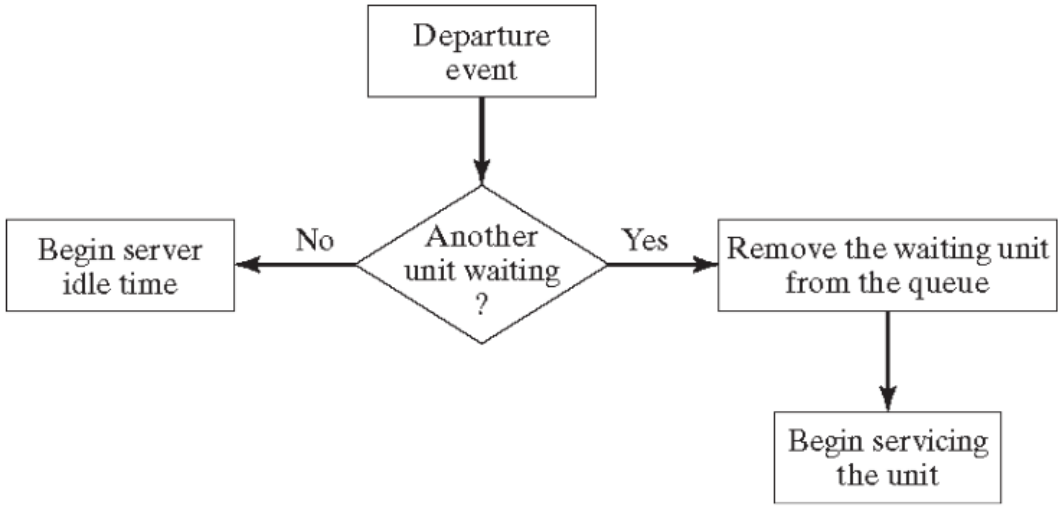
- The Single-Server problem has two types of events: arrivals and departures.
- To simulate using next-event time advance, we need to keep track of events in chronological order
- If you were to implement this scenario using a programming language, what data structure would be appropriate for Future Events List?

Modeling Events

- Arrival of a new customer



- Departure of a served customer



Pseudo-Code for Single-Server Simulation

```
inits: wait_queue ← {}, server ← idle, clock ← 0, FEL ← {}
    generate an arrival event and add it to FEL, customer_count = 1
while FEL not empty:
    remove event from FEL
    update clock to event's time
    if it's an arrival event then call model_arrival()
    else call model_departure()
print_stats()

define model_arrival():
    if customer_count < max_customers // or clock < max_time
    then schedule next arrival event and add to FEL; customer_count++;
    if server busy then add current customer to wait queue
    else server ← busy; schedule current customer's departure event and add to FEL

define model_departure():
    gather stats for the customer about to depart // done serving this customer
    if wait_queue is empty then server ← idle
    else remove customer from wait_queue, schedule their departure event and add to FEL
```