Basic Concept

- Build a computational model of the computer system
- Write a program that executes the model
- Monitor performance metrics in the simulated “system”.

Simulation Methods

Computer Systems Performance Modeling and Evaluation

Department of Electrical and Computer Engineering
Shiraz University
Fall 1388
Reza Azimi

[Some of the Slides are based on Raj Jain’s]
When Should We Use Simulation?

- The system is not available
  - Not built yet
  - Too much cost or logistics to directly measure the system performance
    - Extreme example: evaluation of certain performance metrics over the entire Internet.
- Direct measurement is not possible
  - Example: when direct measurement requires a specific type of hardware monitor which is not available
- Analytical modeling is too complicated
  - We’ll talk about this later.

Basic Terminology

- **State Variables**
  - The variables whose values define the state of the system
  - Examples:
    - The length of the queue for CPU scheduling
    - The number of in-flight packets for a network simulator
  - You can restart simulation by re-initializing the state variables

- **Event**
  - A change in the system state.
  - Example:
    - Job arrival
    - Packet transmission
    - Instruction fetched
Types of Models

- **Continuous Time**
  - System state is defined in all time
  - Examples:
    - Number of jobs in the queue
    - Number of users trying to access a shared resource simultaneously

- **Discrete Time**
  - System state is only defined in some instants
  - Example:
    - The number of in-flight instructions in a CPU (it can only change at each clock cycle, not at any moment).

- **Continuous State**
  - State variables are continuous
  - Examples:
    - Location of a mobile user (in a wireless network).

- **Discrete State (Discrete Event)**
  - State variables are discrete
  - Examples:
    - Queue length
    - Number of active nodes in a cluster
Types of Models

<table>
<thead>
<tr>
<th></th>
<th>Continuous State</th>
<th>Discrete State</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Continuous Time</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ad-hoc wireless networks</td>
<td>CPU Scheduling</td>
</tr>
<tr>
<td><strong>Discrete Time</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Day-to-day power consumption of a system</td>
<td>Micro-architecture Operation</td>
</tr>
</tbody>
</table>

- **Deterministic**
  - The next state is known with certainty
  - The result of the repetitions of the simulation is the same for a set of input
  - Example:
    - Simulation of a CPU

- **Probabilistic**
  - The next state is not certainly known
  - Each repetition may produce different results.
  - Example:
    - Movement of mobile users in an wireless network.
Types of Models

- **Linear**
  - The output is a linear function of the input

- **Nonlinear**
  - The output is a nonlinear function of the input.
  - Computer systems often have a nonlinear behavior
    - Example:
      - Input: Load/Store instructions
      - Output: Cache misses

- **Open**
  - The external input to a model is independent of the output
  - Example:
    - An online store system

- **Closed**
  - There is no external input, jobs circulate in the model
  - Example:
    - A batch system with a fixed number of jobs
Types of Models

- **Stable**
  - The model settles to a steady state after a while
  - Example:
    - Average CPU utilization of an application

- **Unstable**
  - The model keeps changing all the time
  - Example:
    - Instructions in the CPU pipeline

Types of Simulation

- Emulation
- Trace-Driven Simulation
- Discrete-Event Simulation
Emulation

- The simulator provides the full functionality of the simulated system
  - You can use it for real purposes, only it’s slower than the real system

- Examples:
  - Functional emulator of a computer system:
    - Bochs: http://bochs.sourceforge.net/
    - QEMU: http://www.qemu.org

Trace-Driven Simulation

- Trace:
  - Record of events in the system ordered by time
  - The input of the simulation

- Examples:
  - SimpleScalar:
    - Cycle-accurate CPU simulator
    - Trace: A sequence of instructions
  - Any other examples?
Discrete-Event Simulation

- **Basic Operation**
  - Events are generated randomly based on some probability distribution
  - State variables keep updated as events occur

- **Example**
  - ns2/ns3 Network Simulator
    - Used to study Internet protocols and large-scale network systems
    - [http://www.nsnam.org](http://www.nsnam.org)

Discrete-Event Simulation Components

- **Event Scheduler**
  - Schedules event X at time T
  - Holds event X for a time interval

- **Simulated Clock: Time Advancing Mechanism**
  - Unit time:
    - At each step time is advanced by a unit
  - Event-driven
    - At each step time is advanced to the occurrence of the next event

- **System State Variables**

- **Event Handler Routines**
  - One per event (examples: job arrival, job departure, etc.)
Advantages of Trace-Driven Simulation

- Easier Validation:
  - Compare the simulation results with the measured

- Less Non-determinism
  - Trace is a deterministic input => Fewer repetitions

- Accurate Workload
  - Better than random input

- Similarity to the Actual Implementation:
  - Trace-driven model is similar to the system

Advantages of Discrete-Event Simulations

- Simplicity
  - Simple to define, implement, and use

- Flexibility
  - Can change the workload characteristics easily with time and equipment
  - Can generate workload of arbitrary size and intensity