ECE560: Computer Systems Performance Evaluation

Lecture #4 – Simulation Techniques

Instructor: Dr. Liudong Xing

Administration Issues (1/31)

- Homework #1
  - Please download problems from course website
  - Due: **February 5, Monday**
- Project proposal due **February 23, Friday**
  - Refer to the Proposal Guidelines

Review of Lecture #3

- Frequently-used terms in experimental designs
  - Response variables, factors (predictors, predictor variables), levels (treatments), primary factors, secondary factors, replication, designs, experimental units, interactions
- Types of experimental designs
  - Simple designs, full factorial designs, fractional factorial designs
- A closer look at 2^k factorial designs
  - Regression equations and sign table methods to quantify the effects of the factors on the system performance

L#3 Example (revisit)

- Study/quantify the impact of memory size and cache size on the performance of a workstation being designed.

Performance in MIPS		
	A: Memory Size	
B: Cache Size	4MB	16MB
1KB	***15***	***45***
2KB	***25***	***75***

- **Method 1: Regression equation**

$$y = q_0 + q_A x_A + q_B x_B + q_{AB} x_A x_B$$

$$\begin{aligned} 15 &= q_0 - q_A - q_B + q_{AB} \\ 45 &= q_0 + q_A - q_B - q_{AB} \\ 25 &= q_0 - q_A + q_B - q_{AB} \\ 75 &= q_0 + q_A + q_B + q_{AB} \end{aligned}$$

q_0=40
q_A=20
q_B=10
q_{AB}=5

- **Method 2: Sign table**

I	A	B	AB	*y*
1	-1	-1	1	15
1	1	-1	-1	45
1	-1	1	-1	25
1	1	1	1	75
160	80	40	20	Total
40	20	10	5	Total/4

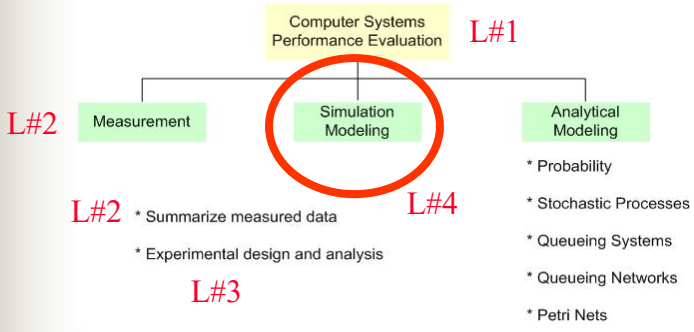
L#3 Review Question

- Please determine the value of ***u***, ***v***, and ***w*** in the table.

I	A	B	AB	*y*
1	-1	-1	1	15
1	1	-1	-1	45
1	-1	1	-1	***u***
1	1	1	1	75
160	80	40	***v***	Total
40	20	10	***w***	Total/4

I	A	B	AB	*y*
1	-1	-1	1	***u***
1	1	-1	-1	50
1	-1	1	-1	40
1	1	1	1	90
200	80	***v***	20	Total
50	20	***w***	5	Total/4

Topics



Simulation

- Basic concepts
- Classifying simulations
- Discrete simulations
- Simulation tools

Definition of Simulation

- Computer simulation modeling is a process of designing a mathematical-logical model of a real system and experimenting with the model on a computer
- Allow inferences to be drawn about the system
  - Without building them if they are only proposed systems
  - Without disturbing them if they are operation systems that are costly or unsafe to experiment with
  - Without destroying them if the object of an experiment is to determine their limits of stress

Systems and Models

- A system
  - A set of interdependent components united to perform a specific function
  - A collection of mutually interacting objects that are affected by outside forces (environment)
- A model
  - An abstraction of a system
  - A description of a system through, for example, computer programs, or some mathematical equations and relations or graphical representations

Model Verification & Validation

- Verify the model
  - The process of determining that the simulation model behaves as intended
- Validate the model
  - The process of determining that the simulation model is a useful or reasonable representation of the system
  - Validation checks the accuracy of the model's representation of the real system

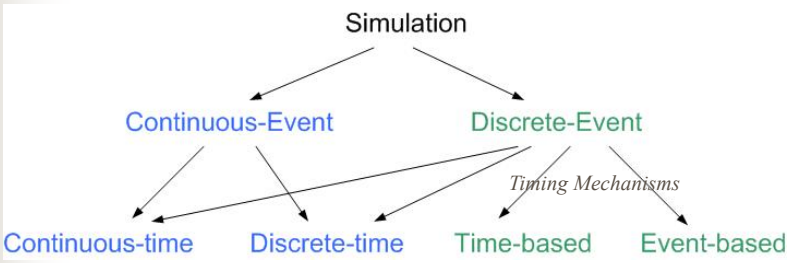
System State Description

- A system is characterized by a set of variables
  - Time is the major independent variable
  - Other variables are function of time and are the dependent variables
- Each combination of variable values represents a unique state/condition of the system
- The manipulation of the variable values simulates movement of the system from state to state
- A simulation experiment involves observing dynamic behavior of a model in accordance with well-defined operating rules designed into the model

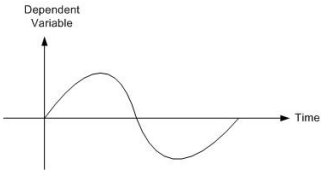
Other Definitions

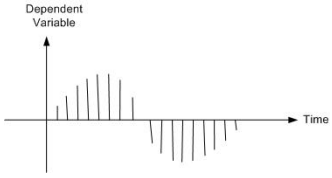
- Events
  - The cause of a state variable change
  - The state changes themselves
- Simulation/simulated time
  - The value of the parameter time used in the simulation program
  - Corresponds to the value of the time valid in the real system
- Run time
  - The time it takes to execute a simulation program

Classifying Simulations

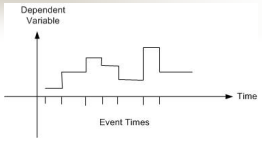


- Continuous-event (continuous) simulations
  - The system state changes continuously over time
  - The dependent variables of the model change continuously over simulated time
  - Example: the level of a reservoir as water flows in and is let out, and as precipitation and evaporation occur
  - A continuous model may be either *continuous* or *discrete* in *time*, depending on whether the values of the dependent variables are available at any point or only at specified points in simulated time





Classifying Simulations (Cont'd)



- Discrete-event (discrete) simulations
  - The state changes discretely with the time
  - The dependent variables of the model change discretely at specified points in simulated time (event times)
  - Example: a manufacturing system with parts arriving and leaving at specific times; machine going down and coming back up at specific times
  - Time variable may be either *continuous* or *discrete*, depending on whether the discrete changes in the dependent variables can occur at any point or only at specified points in simulated time

Classifying Simulations (Cont'd)

- Mixed continuous-discrete models
  - May have elements of both continuous and discrete changes in the same model
  - Example: a refinery with continuously changing pressure inside vessels and discretely occurring shutdowns

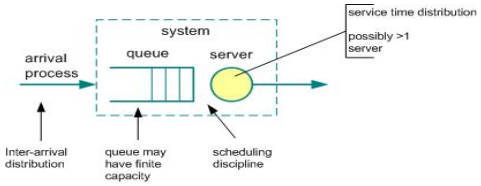
Agenda

- √ Basic concepts /definition
- √ Classifying simulations
- **Discrete simulations**
- Simulation tools

Discrete Simulations

- Based on the timing mechanism (how to advance the simulated time)
  - Time-based: time advanced by a constant step
  - Event-based: time controlled by the occurrence of next events

An Illustrating Example: Queueing Systems

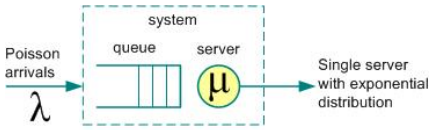


- Arrivals to an empty queue get immediate service
- Arrivals to a busy system are held in the queue until server is free
- Arrival rate is typically drawn from a r.v. distribution, e.g. Poisson with a rate λ
- Service rate is computed using the rate of processing for the device and is in units per time and is often denoted by μ

Applications of Queueing Systems

- Supermarket checkout line
- Bank teller line
- Batch jobs waiting on a CPU
- Traffic lights
- Planes to take off or land
- Airline reservation system
- Read/write requests to a disk controller

M/M/1 Queueing System



- First M: Poisson arrivals with a rate λ
- Second M: Exponential service times with a mean of $1/\mu$, so μ is the average service rate
- 1 server
- An infinite length buffer/queue
- Problem:
  - Determine the average number of jobs in service
  - Determine the average number of jobs in the queue

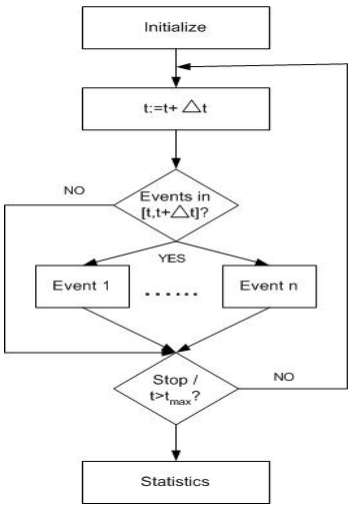
A Specific Example of M/M/1

- Consider a storage system with one disk drive and a queue. The I/O requests arrive to the storage system at the rate of 10 requests per second with Poisson pattern. The time to service an I/O request at the disk drive is exponentially distributed with a mean of 90 milliseconds.
- Problem:
  - Determine the average number of I/O requests in service
  - Determine the average number of I/O requests waiting in the queue

Time-based (synchronous) simulation

- Time advances in constant steps Δt
- The number of events that happened in the interval [t, t+Δt] may change from time to time
- Δt is assumed to be sufficiently small
- After each time advance, check if any events have happened in [t, t+Δt]
  - If so, the events will be executed: the state will be changed according to these events
- When *t* rises above some maximum, simulation stops

Time-Based Discrete Simulation (Cont'd)



Time-Based Discrete Simulation (Cont'd)

- A time-based simulation program for M/M/1
  - Arrival rate λ
  - Service rate μ
  - **Two state variables**

    N_s: # of jobs in service (0/1)

    N_q: # of jobs queued (>=0)

    $N_q>0 \rightarrow N_s=1$
- *Aim*: to generate a list of time-instances and state variables at these instances
- *Program*:
  - Pseudo-code (handout)
  - *draw(p):* evaluates to *true* with probability *p* and to *false* with probability *1-p*

```
1.  input(λ, μ, t_max)
2.  t := 0
3.  N_s := 0; N_q := 0
4.  while t < t_max
5.  do
6.      t := t + Δt
7.      if draw(λ · Δt) then N_q := N_q + 1
8.      if N_s = 1
9.      then if draw(μ · Δt)
10.         then if N_q > 0
11.             then N_q := N_q − 1
12.             else N_s := 0
13.     if N_s = 0 and N_q > 0
14.     then N_s := 1; N_q := N_q − 1
15.     writeln(t, N_q, N_s)
16. od
```

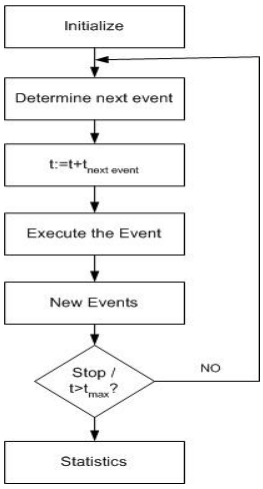
Event-based (asynchronous) simulation

- Simulation is controlled by the occurrence of next events
- Time steps are of varying length such that there is always exactly one event in every time step
- Whenever an event occurs this causes new events to occur in the future
  - Example: event of the arrival of a job at a queue causes the system state to change, will also cause the event that the job is taken into service in the future

Event-Based Discrete Simulation (Cont'd)

- All future events are gathered in an *ordered event list*
  - Head contains the next event to occur and its occurrence time
  - Tail contains the future events in the occurrence order
  - Whenever the first event is simulated/processed, take it from the list and update simulation time accordingly
  - New events that may be created during the simulation of the head event are inserted in the event list
  - The new head event is processed……

Event-Based Discrete Simulation (Cont'd)



Event-Based Discrete Simulation (Cont'd)

- An event-based simulation program for M/M/1
  - Arrival rate λ
  - Service rate μ
  - **Two state variables**
    N_s: # of jobs in service (0/1)
    N_q: # of jobs queued (>=0)
    $N_q>0 \rightarrow N_s=1$
  - **Two next event variables**
    narr: time of the next arrival
    ndep: time of the next departure
- *Aim*: to generate a list of events times, and the state variables at these instances
- *Program*:
  - Pseudo-code (handout)
  - *negexp(λ):* generates a realization of a r.v. with negative exponential distribution with rate λ

```
1.  input(λ, μ, t_max)
2.  t := 0
3.  N_s := 0; N_q := 0
4.  while t < t_max
5.  do
6.      if N_s = 1
7.      then narr := negexp(λ)
8.           ndep := negexp(μ)
9.           if ndep < narr
10.          then t := t + ndep
11.               if N_q > 0
12.               then N_q := N_q - 1
13.               else N_s := 0
13.          else t := t + narr
14.               N_q := N_q + 1
15.     else narr := negexp(λ)
16.          t := t+ narr
17.          N_s := 1
18.     writeln( t, N_q, N_s)
19. od
```

Simulation Tools

- A collection of modeling and simulation resources on the Internet:
  http://home.ubalt.edu/ntsbarsh/simulation/sim.htm

- AweSim: a general-purpose simulation system, supporting
  - Model building
  - Analysis of models using simulation
  - The presentation of simulation results using animations, reports, and graphs
  - References:
    - A. Pritsker & J. O'Reilly, "Simulation with Visual Slam and AweSim (2nd Edition)", Wiley, John & Sons, 1999
    - *Introduction to AweSim by J. Jean O'Reilly and William R. Lilegdon*
    - *AweSim: The Integrated Simulation System by A. Alan B. Pritsker and J. Jean O'Reilly*

Summary of Lecture #4

- Applications of simulation modeling
- Basic concepts /definition of simulations
  - System, model, simulation definition, system state description, event, simulation time, run time
- Classification
  - According to state space
    - Discrete vs. continuous event simulations
  - According to time evolution
    - Discrete vs. continuous time simulations
  - Discrete simulations in terms of time advancing mechanisms
    - Time-based, event-based
- Simulations tools

Things to do

- Homework #1 due **Feb. 5, Monday.**
- Project proposal due **Feb. 23, Friday.**

Next Topic

- Probability Theory (review)