Final Exam for *Simulation* (CIS 4930)

Summer 2009

>>> SOLUTIONS <<<

Welcome to the Final Exam for *Simulation* (CIS 4930). Read each problem carefully. There are ten required problems (each worth 10 points). There is also an additional extra credit question worth 10 points. You may have with you a calculator, pencils and/or pens, erasers, blank paper, and one 8.5 x 11 inch "formula sheet". On this formula sheet you may have anything you want (definitions, formulas, homework answers, old exam answers, etc.) as **handwritten by you in pencil or ink** on both sides of the sheet. Photocopies, scans, or computer generated and/or printed text are not allowed on this sheet. Note to tablet PC users – you may **not** print-out your handwritten text for the formula sheet. You have 120 minutes for the exam. **Please use a separate sheet of paper for the answer to each question**. Good luck and be sure to show your work! A table of T scores is included in this exam for any problem that may ask you to calculate a confidence interval.

Problem #1 2 pts for (a), 2 pts for (b), 3 pts for (c), and 3 pts for (d).

Answer the following question regarding the basics of systems and performance modeling.

a) What is a system?

A system is a set of interacting or interdependent entities forming an integrated whole.

b) What is performance?

Performance is the quantitative measure of a system.

c) What is a model, what is the goal of a model, and why do we build models?

"A model is a representation (physical, logical, or functional) that mimics another object under study" (Molly). The goal of a model is to predict some behavior of an actual object of system. Models are often cheaper, easier, faster, and safer than studying real systems.

d) What is computer simulation?

"Computer simulation is the discipline of designing a model of an actual or theoretical physical system, executing the model on a computer, and analyzing the execution output." (Fishwick).

Problem #2 2 pts for (a), 2 pts for (b), 6 pts for (c) (1 pt for mean, 1 pt for variance and 2 pts for each plot).

Answer the following question regarding the basics of probability theory as they apply to performance modeling.

a) What is the difference between a permutation and a combination?

In a permutation order matters, in a combination order does not matter.

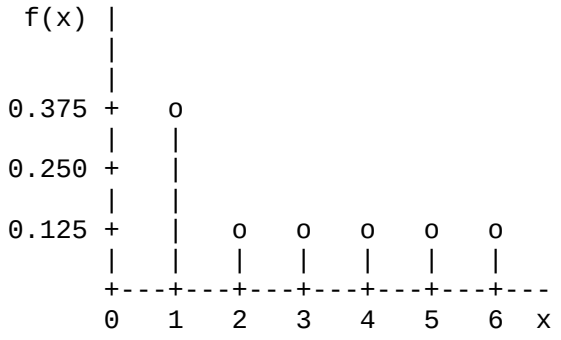
b) What is a random variable?

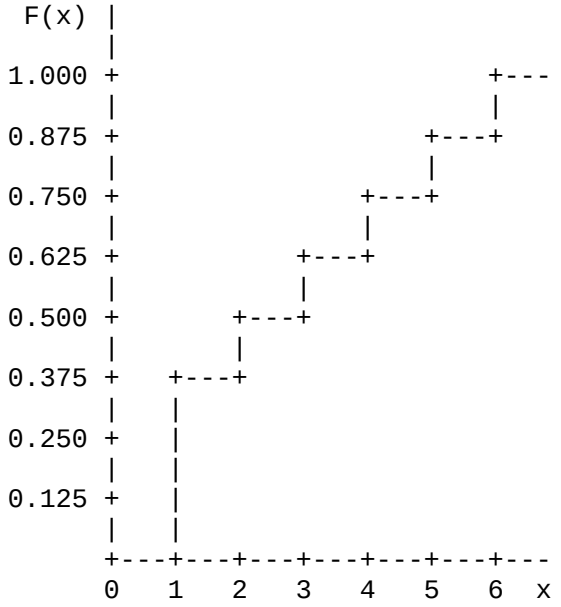
A random variable is a function that maps a real number to every possible outcome in the sample space.

c) Consider an unfair six-sided die where the probability of rolling a "1" is three times the probability of rolling any other value. All other values have the same probability of occurring. Let X be the random variable that takes on values 1 through 6 for our unfair die. Plot the pmf and CDF of X. Determine the mean of X.

Pr[X = 1] = 0.375, Pr[X = 2, 3, 4, 5, and 6] = 0.125

E[X] = (0.375)(1) + (0.125)(2) + (0.125)(3) + (0.125)(4) + (0.125)(5) + (0.125)(6) = 2.875





Problem #3 5 pts for (a) (roughly 1 pt per sub-answer), 5 pts for (b) (2 pts for PK, 3 pts for set-up).

Answer the following questions about queuing theory.

a) For an M/M/1 queue we know that the mean number of customers in the system (L) is equal to the utilization divided by one minus the utilization. Using basic laws and relationships, derive the mean wait in the system (W), the mean number of customers in the queueing area (Lq), and the mean wait in the queuing area (Wq) as a function of arrival rate and service rate. For full credit, your expressions need to be simplified.

We know that $L = \rho/(1-\rho)$ where $\rho = \lambda/\mu$ where $\lambda =$ customer arrival rate and $\mu =$ customer service rate. Little's Law states that $L = \lambda W$. We also know what $Lq = L - \rho$ and $Wq = W - (1/\mu)$. So, we now have $W = 1/(\mu - \lambda)$, $Wq = \lambda/((\mu - \lambda)\mu)$, and $Lq = \rho^2/(1-\rho)$.

b) Consider a single server queue with Poisson arrivals (rate λ) and a uniformly distributed service time (with minimum value a seconds and maximum value b seconds). Solve for the mean number of customers in the system (L). You do not need to simplify your expression for L.

This is an M/G/1 queue where the G is uniform with $\mu = \frac{1}{2}(a+b)$ and $\sigma^2 = \frac{1}{12}(b-a)^2$ and so $Cs = \frac{\sqrt{1/12}\cdot(b-a)}{(1/2)(a+b)}$.

We also have that $\rho = \frac{2\lambda}{a+b}$ Using the PK formula (for M/G/1) we have $L = \rho + \frac{\rho^2(1+Cs^2)}{2(1-\rho)} = \frac{2\lambda}{a+b} + \frac{\left(\frac{2\lambda}{a+b}\right)^2\left(1+\left(\frac{\sqrt{1/12}\cdot(b-a)}{(1/2)(a+b)}\right)^2\right)}{2\left(1-\frac{2\lambda}{a+b}\right)}$ (simplification is not necessary).

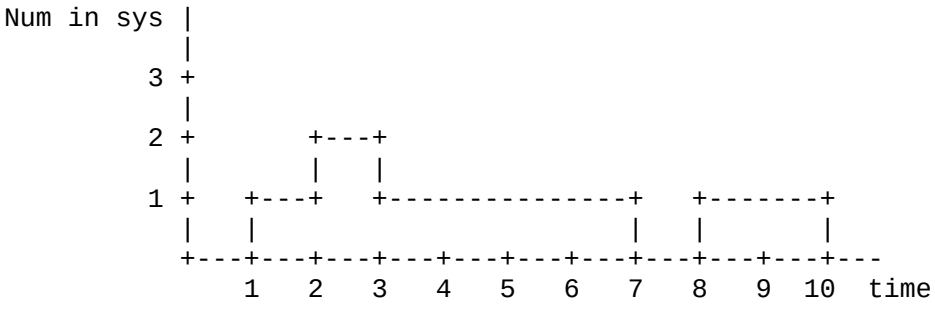
Problem #4

4 pts for graph and 1 pt for each of six measures.

Consider the following single-server queueing system from time = 0 to time = 10 sec. Arrivals and service times are:

- Customer #1 arrives at t = 1 second and requires 2 seconds of service time
- Customer #2 arrives at t = 2 second and requires 2 seconds of service time
- Customer #3 arrives at t = 5 seconds and requires 2 seconds of service time
- Customer #4 arrives at t = 8 seconds and requires 2 seconds of service time

Solve for system throughput (*X*), total busy time (*B*), mean service time (*Ts*), utilization (*U*), mean system time (delay in system) (*W*), and mean number in the system (*L*). Show your work to receive full credit.



X = C/T = 4/10 = 0.4 cust/sec
B = 8 seconds (by inspection)
Ts = B/C = 8 / 4 = 2.0 sec / cust
U = B/T = 8/10 = 80%
W = Sum of w / C = 9 / 4 = 2.25 sec
L = Sum of w / T = 9 / 10 = 0.9 cust

Problem #5

5 pts for Monte Carlo framework, 5 pts for correct solution.

Write a Monte Carlo simulation to model a biased coin as follows. When flipped if the coin show tails it has a 50% chance of tails or heads on the next flop. However, if the coin shows heads then it has 75% chance of showing heads again on the next flip. The Monte Carlo simulation should determine the probability of a head showing. You may assume that you have access to a function rand_val() that returns uniform(0.0, 1.0).

```
int main()
{
  double   head_prob;           // Probability of flipping a head
  double   heads, tails;        // Counters for heads and tails
  int      i;                   // Loop counter

  rand_val(1);
  head_prob = 0.50;
  heads = tails = 0;
```

```
  for (i=0; i<NUM_ITER; i++)
  {
    if (rand_val(0) < head_prob)
    {
      heads++;
      head_prob = 0.75;
    }
    else
    {
      tails++;
      head_prob = 0.50;
    }
  }

  printf("Pr[head] = %f \n", (double) heads / (heads + tails));

  return(0);
}
```

Problem #6

1 pt for C function library, 1 pt for process oriented, 1 pt for each of facility, storage, event, mailbox, table, reporting and random number streams, and 1 pt for run length control.

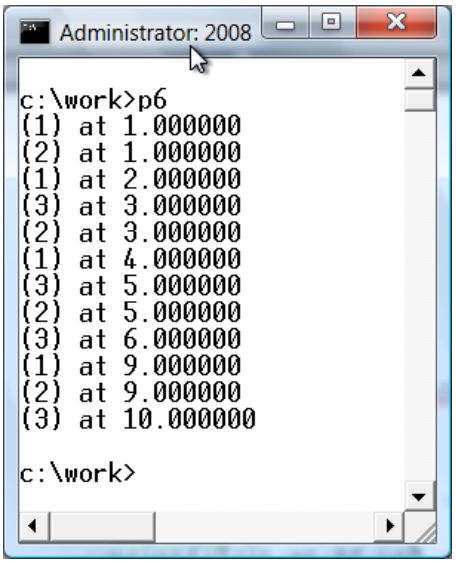
Give an approximately 100 to 150 word overview or description of what is CSIM. You will be graded for completeness (i.e., it is not sufficient to just say "CSIM is a simulation language").

CSIM is a commercial C function library from Mesquite Software for developing process-oriented discrete event simulation models. CSIM was originally developed in the 1980s. A CSIM process models a system component and it can be active, holding (time moves forward), or waiting. CSIM processes run independently. CSIM supports constructs for facilities (entities that do work), storage (entities that can be allocated and de-allocated), events (process synchronization), mailboxes (passing messages between processes), tables (collecting and summarizing data), reporting, and random number streams. Reporting includes standard end-of-run reports and inspector functions for use during simulation runtime and at the end-of-run. CSIM also supports automatic run length control (or stopping) using confidence intervals based on a batch-means method.

Problem #7

Roughly 1 pt for each correct line. Allow for any ordering of simultaneous events.

Appendix A contains a CSIM program. What is the output from this program?



Problem #8

5 pts for (a) (use of table 2 pts, insertion of record() 2pts and D/M/1 is 1 pt), 5pts for (b) (2 for create(), 1 for loop, 2 for hold() and record()).

Answer the following CSIM-related questions.

a) Appendix B contains a CSIM program that models a single server queue. Add a CSIM table to the program to collect data on the number of customers in the system as seen by an arriving customer. What kind of queue (identify using Kendall notation) does this CSIM program model?

The solution is marked in Appendix B. This program models a D/M/1 queue.

b) Write a CSIM process that will periodically (say, every DELAY seconds – DELAY is a globally defined constant value) collect data on how many customers are queued for a CSIM facility named Server (the facility Server has been globally defined). The collected data should be recorded in a CSIM table named Cust_table (Cust_table has been globally defined).

```
void sample()
{
  create("sample");

  while(1)
  {
    hold((double) DELAY);
    record((double) qlength(Server), Cust_table);
  }
}
```

Problem #9

5 pts for identify keying components (1 pts for each component), 3 pts for basic structure, 2 pts for cache hit and else of three holds).

Describe how to model a disk drive unit. Sketch-out a CSIM process for a disk drive that captures a "pretty good" level of detail for modeling reading a file from the disk drive unit. It is important to identify all the components and/or functionality of a disk drive unit that may affect performance.

Key aspects of disk drive performance are the 1) cache hit probability, 2) head seek time, 3) disk spin time, and 4) actual disk (also cache) access rate. The key performance metric of interest is the response time to read a file. A CSIM process for a disk drive read model looks like:

```
void disk(double file_size, double org_time)
{
  create("disk");

  // Process the file read request
  reserve(Disk_facility);
  if (uniform(0.0, 1.0) < Prob_hit)
    hold(file_size / Read_rate1);     // Read the file bytes from cache
  else
  {
    hold(uniform(0.0, Seek_time));     // Move the head
    hold(uniform(0.0, Spin_time));     // Spin the disk to position
    hold(file_size / Read_rate2);      // Read the file bytes from disk
  }
  release(Disk_facility);

  // Record the response time
  record((clock - org_time), Resp_table);
}
```

Where Prob_hit, Read_rate1, Read_rate2, Seek_time, and Spin_time are all global variables that have been defined and declared elsewhere previous to disk() be invoked. In this model we used probability distributions for cache hit, head movement time, and disk spin time. An even more detailed model would have a "real" cache and would completely model head movement based on the head seek algorithm actually used in the disk drive. The disk location would also be modeled.

Problem #10

6 pts for formulas, 1 pt for each correctly computed value.

Below are some sample means taken from independent replications of a simulation model. What is the 95% confidence interval (CI)? What is the accuracy of the CI? A T-score table is given in Appendix C.

10
12
15
15
15
17

$$\bar{Y} = \frac{1}{6} \cdot (10 + 12 + 15 + 15 + 15 + 17) = 14$$

$$S = \sqrt{\frac{1}{6-1} \cdot \left[(10-14)^2 + (12-14)^2 + \ldots + (17-14)^2 \right]} = 2.5298$$

$t_{\alpha/2} = 2.57$ (for 95% confidence we use $\alpha = 0.05$ and N = 6 degrees of freedom, so we use the N - 1 = 5 row)

$$H = 2.57 \cdot \left(\frac{2.5298}{\sqrt{6}} \right) = 2.6543$$

Thus, the population mean (μ) is between 11.346 and 16.6543 with 95% confidence. The accuracy of the CI is 2.6543/14 = 19%.

Extra Credit:

4 pts for two indexes, 4 pts for getting length, 2 pts for picking queue including load balancing.

Write a CSIM process that models the following load balancer. Consider a single load balancer where all customers arrive. There are N queues. A customer can be sent to any of the N queues. For an arriving customer, randomly select two queues out of the N. Send the customer to the queue (of the two selected) with the smallest number of customers in the system. Ties need to be broken fairly. The prototype for the queue process (that you need not write) is `void queue[i](double org_time);` Each queue has a facility `Server[i]` in it. The index `i` ranges from 0 to $N - 1$.

```
void loadBalancer()
{
  int index1, index2;    // Index for queues
  int len1, len2;        // Number in system for selected queues

  // Select two queues
  index1 = random_int(0, (N-1));
  index2 = random_int(0, (N-1));

  // Get the number in the system for the two selected queues
  len1 = qlength(Server[index1]) + num_busy(Server[index1]);
  len2 = qlength(Server[index1]) + num_busy(Server[index2]);

  // Forward customer to the shortest queue, break ties fairly
  if (len1 < len2)
    queue[index1](double clock);
  else if (len2 < len1)
    queue[index2](double clock);
  else
  {
    If (uniform((double) 0.0, (double) 1.0) < 0.5)
      queue[index1](double clock);
    else
      queue[index2](double clock);
  }
}
```

Appendix A – Source code for problem #7

```
// CSIM model for problem #7 on the final exam (summer 2009)
#include <stdio.h>
#include "csim.h"

#define SIM_TIME 1.0e6

FACILITY Server;

void generate(void);
void queue1(double service_time);

void sim(void)
{
  create("sim");

  Server = facility("Server");

  generate();
  hold(SIM_TIME);
}

void generate()
{
  create("generate");

  hold(1.0);
  queue1(2.0);
  hold(1.0);
  queue1(2.0);
  hold(2.0);
  queue1(1.0);
  hold(5.0);
  queue1(1.0);
}

void queue1(double service_time)
{
  create("queue1");

  printf("(1) at %f \n", clock);
  reserve(Server);
  printf("(2) at %f \n", clock);
  hold(service_time);
  release(Server);
  printf("(3) at %f \n", clock);
}
```

Appendix B – Source code for problem #8

```
// CSIM model for problem #8 on the final exam (summer 2009)
#include <stdio.h>
#include "csim.h"

#define SIM_TIME 1.0e6

FACILITY Server;
TABLE    Cust_table;

void generate(double lambda, double mu);
void queue1(double service_time);

void sim(void)
{
  double lambda, mu;

  create("sim");

  Server = facility("Server");
  Cust_table = table("Customer table");

  lambda = 0.9;
  mu = 1.0;

  generate(lambda, mu);
  hold(SIM_TIME);

  report();
}

void generate(double lambda, double mu)
{
  double   interarrival_time;
  double   service_time;

  create("generate");

  while(1)
  {
    interarrival_time = 1.0 / lambda;
    hold(interarrival_time);
    service_time = exponential(1.0 / mu);
    queue1(service_time);
  }
}

void queue1(double service_time)
{
  create("queue1");

  record((double) (qlength(Server) + num_busy(Server)), Cust_table);

  reserve(Server);
  hold(service_time);
  release(Server);
}
```

Appendix C – T score table

Selected values of $t_{\alpha/2;N-1}$

	$\alpha/2 = 0.05$	$\alpha/2 = 0.025$
N – 1	**t**	**t**
4	2.13	2.78
5	2.02	2.57
6	1.94	2.45
7	1.90	2.37
8	1.86	2.31
9	1.83	2.26
10	1.81	2.23
11	1.80	2.20
12	1.78	2.18