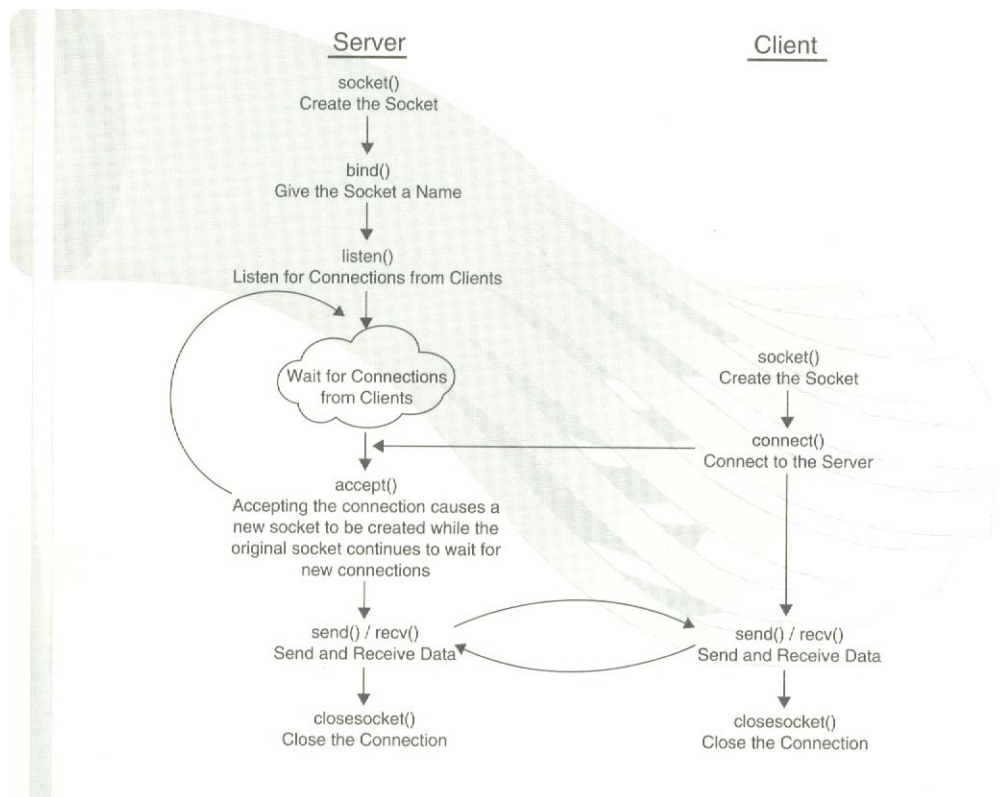


>>> SOLUTIONS <<<

Welcome to the comprehensive Final Exam for *Computer Networks*. Read each problem carefully. There are 10 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 (iPad, etc.) users – you may **not** print-out your handwritten text for the formula sheet. You have 120 minutes for this exam. **Please use a separate sheet of paper for the answer to each question.** Good luck and be sure to show your work!

Problem #1 Each box and flow worth about 1 point.

Sketch the flowcharts of a TCP/IP server and client showing key sockets functions and interactions (or flows) between the two flowcharts. You need not list the function arguments, you just need to give the function names (e.g., bind(), etc.).



Copied from Arthur Dumas, Programming Winsock, SAMS Publishing, 1995.

Problem #2 Identifying delays is 4 points. U formula is 6 points. If claim of negligible is made must explain why.

Derive U (utilization) for a Stop-and-Wait protocol. Assume that packets are never lost.

For a packet-ACK exchange we have the following delays, t_{prop} , t_{xmit} , t_{proc} , t_{prop} , t_{ack} , and t_{proc} . Of these delays, t_{xmit} is useful (data transfer), the rest are overhead and thus:

$$U = \frac{t_{xmit}}{t_{xmit} + 2 \cdot t_{prop} + 2 \cdot t_{proc} + t_{ack}}$$

For a link that is “wide area” (many miles in distance where $t_{prop} > 10 \mu s$), for modern processors that can process thousands of instructions per millisecond, and for ACK packets 10x smaller than data packets, t_{proc} and t_{ack} are negligible and thus U can be approximated as:

$$U = \frac{t_{xmit}}{t_{xmit} + 2 \cdot t_{prop}}$$

Problem #3 Each section identified is 2 points. Note on “analytical modeling” is 2 points.

Briefly (in 100 words or less) summarize Kleinrock’s 1993 *IEEE Proceedings* paper “On the Modeling and Analysis of Computer Networks”.

This paper is a “...landscape of analytic models for computer network performance evaluation.” The paper has four major topics, which are, 1) delay analysis of networks using the M/M/1 model and Kleinrock’s independent assumption, 2) networks as a design problem using classic minimization techniques, 3) networks as a control problem (routing and flow control) introducing the measure of power for flow control methods, and 4) a discussion of how gigabit networks are different than previous lower-speed networks (notably more packets are in flight than buffer size making flow control difficult) introducing the concept of latency and bandwidth limited networks.

Problem #4 Identifying number of trails is 4 points. Correct listing of experiments is 6 points.

Given factors A, B, and C where factors A and B have 2 levels each and factor C has 3 levels, show a full-factorial experimental design. How many experiment trials must be run if for each factor level combination a total of 30 replications are needed?

For a full-factorial design we will have $2 \times 2 \times 3 = 12$ experiments, with 30 trials per experiment we will have 360 trials to run. The experimental design (showing here factors and factor levels is):

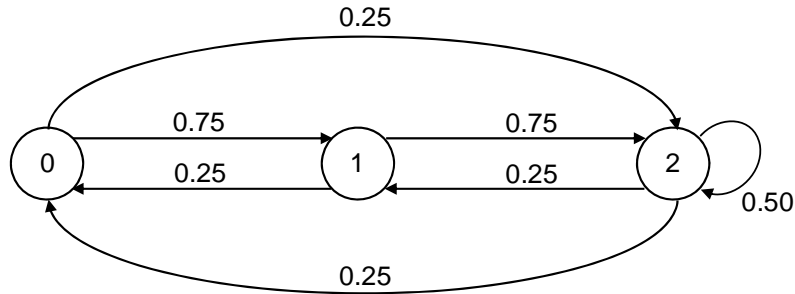
- A1, B1, C1
- A1, B1, C2
- A1, B1, C3
- A1, B2, C1
- A1, B2, C2
- A1, B2, C3
- A2, B1, C1
- A2, B1, C2
- A2, B1, C3
- A2, B2, C1
- A2, B2, C2
- A2, B2, C3

Problem #5 Markov chain is 4 points. Set-up of equations is 4 points. Correct solution is 2 points.

For the following P matrix sketch the corresponding Markov chain. Solve for the steady state probabilities for the three states.

$$P = \begin{bmatrix} 0 & 0.75 & 0.25 \\ 0.25 & 0 & 0.75 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}$$

The Markov chain is:



We can solve for the steady state probabilities (π_0 , π_1 , and π_2) by solve for three equations derivable from the P matrix, which are:

$$\begin{aligned} \pi_0 &= 0.25\pi_1 + 0.25\pi_2 \\ \pi_1 &= 0.75\pi_0 + 0.25\pi_2 \\ \pi_2 &= 0.25\pi_0 + 0.75\pi_1 + 0.5\pi_2 \end{aligned}$$

To solve, we need to break linear dependence so we replace the third equation with the conservation equations as:

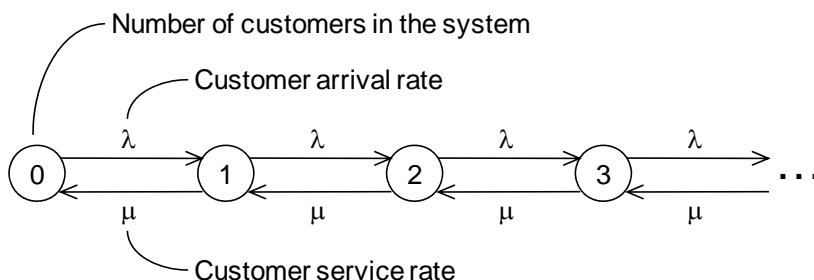
$$\begin{aligned} \pi_0 &= 0.25\pi_1 + 0.25\pi_2 \\ \pi_1 &= 0.75\pi_0 + 0.25\pi_2 \\ 1 &= \pi_0 + \pi_1 + \pi_2 \end{aligned}$$

The solution to these three equations in three unknowns is $\pi_0 = 0.20$, $\pi_1 = 0.28$, and $\pi_2 = 0.52$. You should recognize this problem as the Kleinrock "hippie problem" presented in class.

Problem #6 Markov chain is 2 points. Set-up for π_i , π_0 , L , and W are 2 points each.

Derive W for M/M/1 starting with a Markov chain.

We first draw the Markov chain for the M/M/1 queue.



We solve the Markov chain for L . Once we have L we can easily find W using Little's Law ($L = \lambda W$). Solving for L , from local balance we write,

$$\begin{aligned}\pi_0 \lambda &= \pi_1 \mu \Rightarrow \pi_1 = \left(\frac{\lambda}{\mu}\right) \pi_0 \\ \pi_1 \lambda &= \pi_2 \mu \Rightarrow \pi_2 = \left(\frac{\lambda}{\mu}\right) \pi_1 = \left(\frac{\lambda}{\mu}\right)^2 \pi_0 \\ \pi_i &= \left(\frac{\lambda}{\mu}\right)^i \pi_0 \text{ for } i = 1, 2, \dots\end{aligned}$$

We can now solve for π_0 with the knowledge that the sum of π_i must equal one,

$$\begin{aligned}1 &= \pi_0 + \sum_{i=1}^{\infty} \left(\frac{\lambda}{\mu}\right)^i \pi_0 \\ \pi_0 &= 1 - \frac{\lambda}{\mu} = 1 - \rho\end{aligned}$$

So, now we have the steady state probabilities as,

$$\pi_i = (1 - \rho) \rho^i$$

The mean then follows directly from the definition of mean,

$$L = \sum_{i=0}^{\infty} i \pi_i = \sum_{i=0}^{\infty} i (1 - \rho) \rho^i = \frac{\rho}{1 - \rho}$$

Now we can solve,

$$W = \frac{\rho}{\lambda} = \frac{1}{\mu - \lambda}.$$

Problem #7

P-K formula is 2 points. 4 points for \bar{x} and \bar{x}^2 . 2 points for algebra for result.

Derive W for M/M/1 starting with the P-K formula.

The P-K formula is,

$$W = \bar{x} + \frac{\lambda \bar{x}^2}{2(1 - \rho)}$$

where \bar{x} is the first moment of the service time and \bar{x}^2 is the second moment of the service time (which is not the same as the first moment squared). For M/M/1 with exponentially distributed service rate μ we know that,

$$\bar{x} = \frac{1}{\mu} \text{ and } \bar{x}^2 = \frac{2}{\mu^2}.$$

Thus,

$$W = \frac{1}{\mu} + \frac{\lambda}{\mu^2(1 - \rho)} = \frac{1}{\mu} + \frac{\rho}{\mu(1 - \rho)} = \frac{1 - \rho}{\mu(1 - \rho)} + \frac{\rho}{\mu(1 - \rho)} = \frac{1}{\mu(1 - \rho)} = \frac{1}{\mu - \lambda}$$

Which is W for M/M/1 as derived from the Markov chain in the previous problem.

Problem #8 Facility 1 is 1 point. Each function is 3 points.

Write a simulation model for an M/M/1 queue using the CSIM function library.

```
#include "csim.h"

#define SIM_TIME 1000000.0

FACILITY Server;
double Lambda;
double Mu;

void generate(void);
void queuel(void);

void sim(void)
{
    create("sim");
    Server = facility("Server");
    Lambda = 1.0;
    Mu = 3.0;
    generate();
    hold(SIM_TIME);
    report();
}

void generate(void)
{
    create("generate");
    while(1)
    {
        hold(exponential(1.0 / Lambda));
        queuel();
    }
}

void queuel(void)
{
    create("queuel");
    reserve(Server);
    hold(exponential(1.0 / Mu));
    release(Server);
}
```

Problem #9 Description (noting 0.5 to 1.0) is 4 points. Each remaining sub-question is 2 points.

Describe the Hurst parameter. Who was Hurst? Who was the first to apply the Hurst parameter to network traffic (and when did this occur)? What is the possibly impact (to capacity planning of networks) of traffic with a large Hurst parameter value?

The Hurst parameter is a measure of self-similarity (or long-range dependence) of a time series. The Hurst parameter varies from 0.50 (completely independent) to 1.0 (completely self-similar). There are various ways to estimate the Hurst parameter for a time series. Hurst was a hydrologist studying the historical record of flooding of the Nile river in order to “capacity plan” the Aswan Dam. Leland et al. in the early 1990s discovered that network traffic is self similar and used the Hurst parameter as a measure of this self similarity. Bursty self similar traffic streams when aggregated or multiplexed together do not “smooth out”... they remain bursty. This means that capacity planning based on Poisson assumptions will underpredict the required bandwidth and/or buffering needed to carry traffic.

Problem #10

Sub-question (a) is 4 points, the remaining two sub-questions are 3 points each.

Answer the following questions regarding WSNs:

- a) What are the necessary and optional components of a WSN node?

Necessary components are: sensing unit, memory (or storage), processor, communications, and power source. Optional components include: power generating unit (solar, vibration, other), location finding (GPS or other), and a means of mobility.

- b) Why is source routing used (for example, as in AODV) in WSNs?

Internet routing protocols (distance-vector and link-state) are too chatty – that is, they generate overhead traffic that would be unacceptable for power-constrained sensor networks. Source routing is “reactive routing” that generates overhead only when a route is needed – AODV is thus intended to generate less overhead traffic than existing Internet protocols.

- c) What are the methods in the second tier (or level) of Anastasi et al. taxonomy for energy conservation in WSNs? Hint: There are three methods in this tier.

Duty cycling, data driven, and mobility.

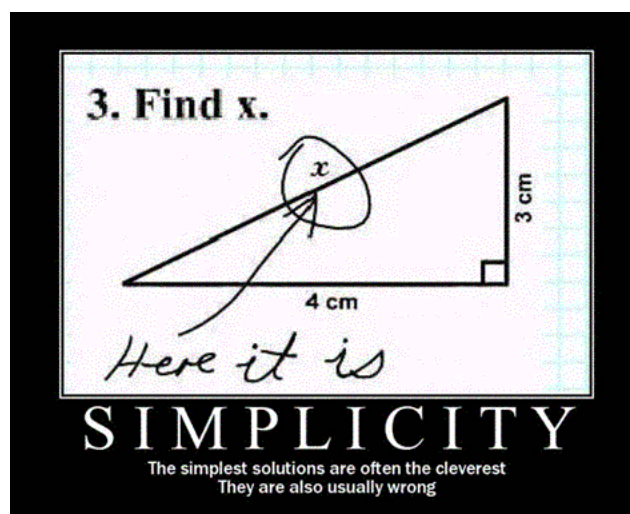
Extra Credit

“What is” is 4 points. The remaining two sub-questions are 3 points each.

Describe the SIP Catcher. What is its purpose? How does it work? Who developed it?

The SIP Catcher is a proxy for an IP phone. The SIP catcher allows the IP phone that it covers (which could be a PC running a SIP-based VoIP telephony application) to sleep, yet still ring on an incoming call. The SIP Catcher “catches” the incoming ring message (called “trying”) and wake-ups the sleeping IP phone before forwarding this message to the IP phone. The SIP Catcher is software running in a broadband home router that is nominally always powered-on in any case. The SIP Catcher was developed by Miguel Jimeno, a past Ph.D. student now at UNINORTE in Colombia.

Humor



From: <http://listverse.files.wordpress.com/2007/12/image008.gif>

I hope that everyone did well 😊