Comprehensive Final Exam for Computer Networks (CNT 6215) Fall 2014

>>> 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

2 points for each sub-question.

Answer the following questions regarding the basics principles and concepts of networks.

a) Sketch the 7-layer OSI reference model. In one sentence, or less, describe the function of (or service provided by) each layer.

```
  +--------------+
7 | Application  | --  Provides access to user applications
  +--------------+
6 | Presentation | --  Provides data independence
  +--------------+
5 |   Session    | --  Manages end-to-end connections
  +--------------+
4 |  Transport   | --  Provides reliable end-to-end data transport
  +--------------+
3 |   Network    | --  Maintains point-to-point connections
  +--------------+
2 |   Data Link  | --  Provides reliable point-to-point data transport
  +--------------+
1 |   Physical   | --  Transmission of bit stream
  +--------------+
```

b) Which layers in the 7-layer OSI reference model are not found in the 5-layer Internet Protocols stack that we used in the class?

The Presentation and Session layers are not found in the 5-layer Internet Protocols stack.

c) Name at least one protocol that corresponds with (or maps to) each layer of the 5-layer Internet Protocols stack.

Some possible answers are as follows. Application: HTTP, FTP. Transport: TCP, UDP. Network: IP. Data Link: Wi-Fi, Ethernet. Physical: Wi-Fi, Ethernet.

d) In his paper "The Design Philosophies of the DARPA Internet Protocols" Clark describes a top-level goal and seven second-level goals of the DARPA Internet Architecture. State the top-level goal.

The top level goal at stated by Clark is "effective technique for multiplexed utilization of existing interconnected networks".

e) State Saltzer's "end to end principle" for the Internet. Give an example of this principle.

The end-to-end principle states that if the required function can only be implemented at end node, then implementing it in the network is not possible. An example of this is that error detection and correction can never be fully done by the network (an error could occur in the last hop to the end node) so should do it at the end nodes (and not in the network).

Problem #2

1 point for each bug (11 bugs, so one free missed bug).

In Appendix A is the source code for an implementation of a TCP server. The implementation contains several significant bugs. Identify the bugs. The program compiles with no warnings or errors.

Problem #3

4 points for (a), 3 points for (b) and (c) each

The distance from Earth to Mars varies between about 35 million miles to 250 million miles. Let's consider the maximum distance – 250 million miles. Assume a 1 Mb/s communications link to Mars and 1500 byte packets. Answer the following questions.

a) Consider a stop-and-wait (SAW) ARQ protocol, what would the maximum achievable link utilization be (say, for sending data from Mars to Earth)?

We know that (frame = packet),

$$U = \frac{t_{fr}}{t_{fr} + t_{prop}}$$

where $t_{fr} = L/R$ (for L = frame length in bit and R = data rate in b/s) and $t_{prop} =$ distance multiplied by propagation rate. So, for this problem $t_{fr} = (1500 \cdot 8)/1000000 = 0.012$ s and $t_{prop} = 5 \cdot 10^{-6} \cdot 250 \cdot 10^{6} = 1250$ s. So,

$$U = \frac{0.012}{0.012 + 1250} = 0.001\%$$ (so about 1 kb/s)

b) Consider a sliding window (SW) ARQ protocol, what window size would be needed to achieve 100% link utilization?

To achieve 100% utilization the "pipe" needs to be kept full. The window size required is then,

$$W = \frac{t_{fr} + t_{prop}}{t_{fr}} = 104168 \text{ frames}$$

c) What is a possible alternative to an SAW or SW ARQ protocol to reliably transfer data from Mars to Earth? Comment on the reliability and efficiency of this alternative.

Forward Error Correction (FEC) is the alternative to ARQ. In FEC redundant data is sent in each packet to enable detection and recovery from any lost packets. When no errors occur, this redundant data is overhead. If too many errors occur (e.g., too many consecutive packets lost) then FEC cannot recover the lost data – that is, FEC has its limits on error detection and recovery.

Problem #4

2 points for (a) and (c) each. 3 points foro (b) and (d) each.

Answer the following questions related to Layer-2.

a) What are the four ways to share a channel?

Frequency Division Multiplexing (FDM), Time Division Multiplexing (TDM), Contention, and Token Passing.

b) Describe the basic ALOHA protocol.

```
Sender:
   Transmit packet when have a packet to send
   If not receive an ACK then resend packet a random time later

Receiver:
   If receive a packet then send an ACK to the sender
```

c) There is a relatively simple way to double the maximum achievable throughput (or utilization) of ALOHA. Describe this method and explain why and how it can double the maximum achievable throughput.

Slotted ALOHA can double the possible throughput from about 18% to about 36%. In Slotted ALOHA sends can only send on the start of a common (global) clock interval. This reduce the vulnerable period by half and thus doubles the achievable throughput.

d) What is the goal of the Ethernet back-off algorithm? Describe (e.g., in pseudocode) the Ethernet back-off algorithm

Let there be Q stations with packets queued for transmission. The goal of the back-off algorithm is to achieve probability of transmission in the next transmission "slot" to be 1/Q. The algorithm cannot explicitly know what Q is, it has to estimate it. The back-off algorithm for Ethernet is Binary Exponential Backoff (BEB) and it is:

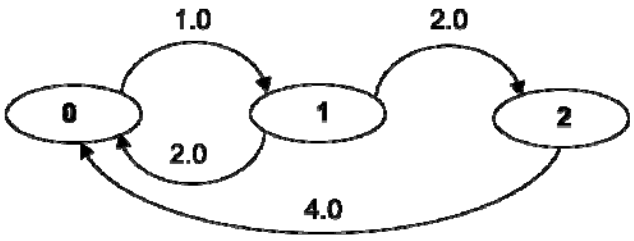
```
while (attempts < backoff_limit)
   k := min(attempts, 10)
   r := rand(0, 2**k)
   Delay := r * slot_time
```

where attempts is reset to zero on a successful transmission and incremented on a collision. The slot_time is equal to the time to transmit a minimum length (64 byte) packet.

Problem #5

5 points for each

Below is a CTMC. Answer the following questions with respect to this CTMC.



a) Give the Q matrix and set-up the equations necessary to solve for the steady-state probabilities (you do not need to solve the equations, just set them up).

$$Q = \begin{bmatrix} -1 & 1 & 0 \\ 2 & -4 & 2 \\ 4 & 0 & -4 \end{bmatrix}$$

$$\begin{aligned} -\pi_0 + 2\pi_1 + 4\pi_2 &= 0 \\ \pi_0 - 4\pi_1 &= 0 \\ \pi_0 + \pi_1 + \pi_2 &= 1 \end{aligned}$$

(Note the last equation to break linear dependence)

b) Convert the Q matrix to a P matrix using uniformization and set-up the equations necessary from the P matrix to solve for the steady-state probabilities (you do not need to solve the equations, just set them up).

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \left(\frac{1}{4}\right)\begin{bmatrix} -1 & 1 & 0 \\ 2 & -4 & 2 \\ 4 & 0 & -4 \end{bmatrix} = \begin{bmatrix} 0.75 & 0.25 & 0.00 \\ 0.50 & 0.00 & 0.50 \\ 1.00 & 0.00 & 0.00 \end{bmatrix}$$

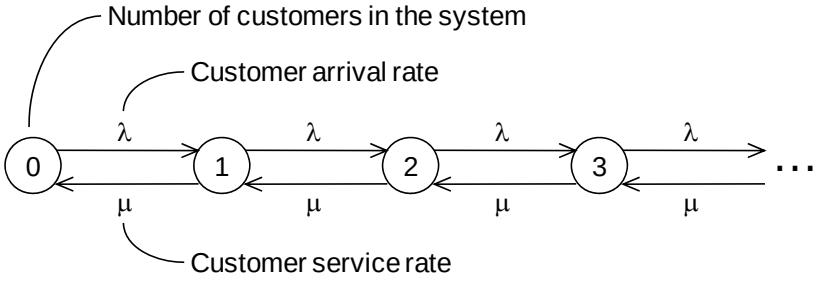
$$\begin{aligned} 0.75\pi_0 + 0.50\pi_1 + 1.00\pi_2 &= \pi_0 \\ 0.25\pi_0 &= \pi_1 \\ \pi_0 + \pi_1 + \pi_2 &= 1 \end{aligned}$$

Problem #6

2 points for DTMC, 2 points for set-up for L, 2 points for L, 4 points for rest

Solve the M/M/1 queue. That is, derive closed-form expressions for π_i (steady state probability for i customers in the system, i = 0, 1, …), L (mean number of customers in the system), W (mean delay in the system), Lq (mean number of customers in the buffer), and Wq (mean wait in the buffer). Simply giving the formulas is not sufficient, you must derive them.

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* = λ*W*). Solving for *L*, from local balance we write,

$$\pi_0\lambda = \pi_1\mu \Rightarrow \pi_1 = \left(\frac{\lambda}{\mu}\right)\pi_0$$

$$\pi_1\lambda = \pi_2\mu \Rightarrow \pi_2 = \left(\frac{\lambda}{\mu}\right)\pi_1 = \left(\frac{\lambda}{\mu}\right)^2 \pi_0$$

$$\pi_i = \left(\frac{\lambda}{\mu}\right)^i \pi_0 \text{ for } i = 1,2,\ldots$$

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

$$1 = \pi_0 + \sum_{i=1}^{\infty}\left(\frac{\lambda}{\mu}\right)^i \pi_0$$

$$\pi_0 = 1 - \frac{\lambda}{\mu} = 1 - \rho$$

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,

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

Having found L, we can now find W from Little's Law ($L = \lambda W$) and also Lq and Wq from

$$L_q = L - \rho$$

$$W_q = W - \frac{1}{\mu}$$

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

$$L_q = \frac{\rho}{1-\rho} - \rho = \frac{\rho^2}{1-\rho}$$

$$W_q = \frac{1}{\mu-\lambda} - \frac{1}{\mu} = \frac{\lambda}{\mu(\mu-\lambda)}$$

Problem #7 2 points for new queue process. 2 points for traffic forwarding. 6 points for rest.

In Appendix B is the source code for a CSIM model of an M/M/1 queue. Modify this program to model tandem queues where customers departing from queue #1 enter queue #2. The arrival rate of customers into queue #1 is λ and the service rates of queue #1 and #2 are μ1 and μ2, respectively. The input parameters are λ, μ1, and μ2. Assume exponentially distributed service time for both queues and Poisson arrivals to the first queue. The simulation should output the mean system delay experienced by arriving customers.

Problem #8

2.5 points each (but, round-up to next full point in any case)

We reviewed four key papers in the area of traffic characterization. These papers were assigned reading for you. The papers were by Jain et al. (1986), Leland et al. (1994), Barford et al. (1998), and Karagiannis et al. (2004). For each paper give the major question(s) addressed and summarize in one or two sentences the findings of the paper.

Jain et al. – Is LAN traffic Poisson? This paper shows that LAN traffic is not Poisson and then proposes a so-called "packet train" model.

Leland et al. – Is LAN traffic self-similar? This paper is a major study of LAN traffic at Bellcore where is is shown the LAN traffic is self-similar with an Hurst parameter of about 0.80. A visual comparison of LAN traffic versus synthetic compound Poisson traffic shows self-similarity (and lack of smoothing) for the LAN traffic of a range of time scales; the compound Poisson traffic is smooth at large time scales.

Barford et al. – Can a better traffic model (for use in a benchmark) exercise a web server more realistically? Web traffic is characterized from traces with the notion of a "User equivalent" and a benchmark built using distributions determined from the traces show a more realistic CPU load.

Karagiannis et al. – Is network traffic self-similar? Summary is that current network traffic can be modeled as Poisson for sub-second time scales and at multiple second time scales piecewise-linear and non-stationary with evidence of LRD.

Problem #9

2 points for each item.

Answer the following questions about sensor networks.

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) What are the protocol stack layers and planes for a WSN node?

The layers are the usual physical, data link, network, transport, and application. The three planes are power management, mobility management, and task management.

c) What is the key challenge (to be addressed by research) for Layer 3 for WSNs?

Existing routing protocols (e.g., RIP, OSPF) are too chatty thus making them energy inefficient. Thus a key challenge is to explore new routing protocols.

d) What are the three main enabling techniques for power management in WSNs (hint: recall the taxonomy in Anastasi et al.).

Duty cycling, Data-driven, and Mobility

e) What is IOT?

IOT = Internet of Things – a focus of future research where many of the things may be wireless sensor nodes.

Problem #10 2 points for each of (a) and (c), 3 points for each of (b) and (d)

Answer the following questions about SDN and NFV.

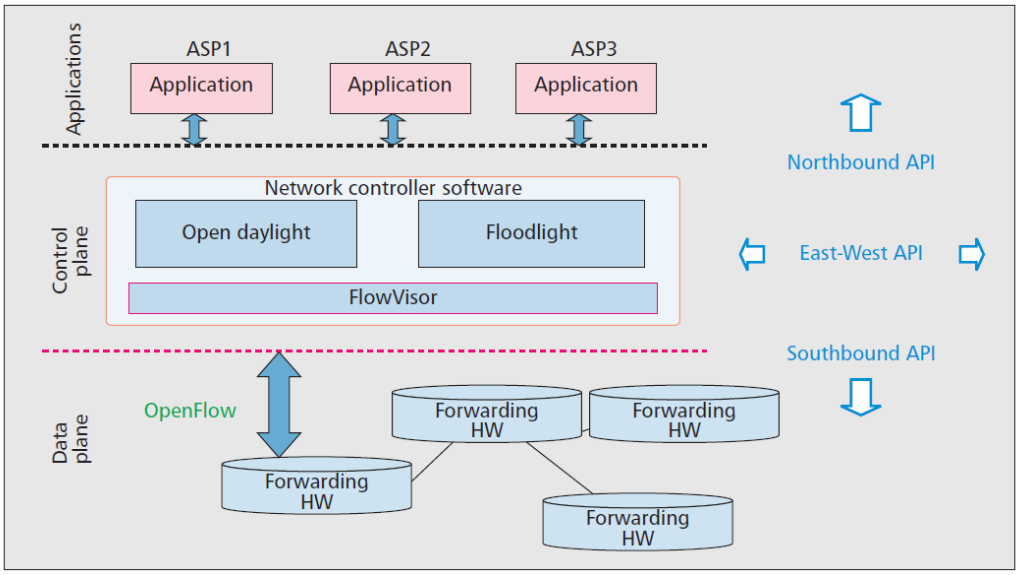
a) What is SDN in one sentence?

"The physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices." (from https://www.opennetworking.org/sdn-resources/sdn-definition)

b) Sketch a picture of network equipment today (label the key components) and then sketch a matching picture of network equipment for an SDN (again, label the key components and also label the key APIs).

A network switch or router today. Closed system. No open source or open APIs.

An SDN network. Note the separated control plan with OpenFlow control of forwarding hardware.



c) What is NFV and what are the motivations for NFV? Give a specific example of a virtualized network equipment.

NFV is Network Function Virtualization and the basic idea is to run network appliance "code" in high-volume servers. The motivations are much the same as for virtualization servers, namely:

- Increase resource utilization
- Reduce hardware cost (CAPEX)

- Reduce energy use and management overhead (OPEX)
- New features and faster upgrades
- Fast spin-up and spin-down of resources

An example of a virtualized network equipment is vSwitch, an open source virtualized Ethernet switch that runs in a Linux server.

d) What are the three components of NFV?

The three components are:

- Virtualized Network Functions (VNF) = Software implementations of network functions that can be deployed on an NFVI
- NFV Infrastructure (NFVI) = all hardware and software components which build up the environment in which VNFs are deployed.
- Network Functions Virtualization Management and Orchestration = the collection of all functional blocks, data repositories, and interfaces through which these functional blocks exchange information for the purpose of managing and orchestrating NFVI and VNFs.

Extra Credit 5 points for the assumption, 5 points for the importance

What is the Kleinrock independence assumption and why is it important?

"The Independence Assumption assumes that the length of a message is chosen independently from the exponential distribution each time it enters a switching node in the computer network!" (Kleinrock, 1993). The significance of this assumption is that it makes analysis of a network of queues (that is, modeling switching nodes in a computer network) trivial by allowing the application of the M/M/1 delay formula independently to each queue. Without the independence assumption the fixed message size would result in correlated delays between queues, which is intractable for analysis.

Appendix A – Source code for Problem #2

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <windows.h>

#define  PORT_NUM   3.14159

int main()
{
  WORD wVersionRequested = MAKEWORD(1,1);
  WSADATA wsaData;
  int                 welcome_s;
  struct sockaddr_in  server_addr;
  int                 connect_s;
  struct sockaddr_in  client_addr;
  struct in_addr      client_ip_addr;
  int                 addr_len;
  char                out_buf[4096];
  char                in_buf[4096];
  int                 retcode;


  welcome_s = socket(AF_INET, SOCK_DGRAM, 0);
  if (welcome_s < 1000){
    printf("*** ERROR - socket() failed \n");
    exit(-1);
  }

  server_addr.sin_family = AF_INET;
  server_addr.sin_port = htons(PORT_NUM);
  server_addr.sin_addr.s_addr = htonl(INADDR_ANY);



  printf("Waiting for accept() to complete... \n");
  addr_len = sizeof(client_addr);
  connect_s = accept(welcome_s, (struct sockaddr *)&client_addr, &addr_len);
  if (connect_s > 0){
    printf("*** ERROR - accept() failed \n");
    exit(-1);
  }

  strcpy(out_buf, "This is a message from SERVER to CLIENT");
  retcode = send(welcome_s, out_buf, (strlen(out_buf) + 1), 0);
  if (retcode < 0){
    printf("*** ERROR - send() failed \n");
    exit(-1);
  }

  retcode = recv(welcome_s, in_buf, sizeof(in_buf), 0);
  if (retcode < 0) {
    printf("*** ERROR - recv() failed \n");
    exit(-1);
  }
  printf("Received from client: %s \n", out_buf);

  retcode = closesocket(welcome_s);
  if (retcode < 0) {
    printf("*** ERROR - closesocket() failed \n");
    exit(-1);
  }

  WSAStartup(wVersionRequested, &wsaData);

  return(0);
}
```

Must be integer

Missing WSAStartup()

Should be SOCK_STREAM

Should be less than zero

Missing bind()

Missing listen()

Should be greater than zero

Should be connect_s

Should be connect_s

Should be in_buf

Need to close the connect_s socket

Should be WSACleanup()

Appendix B – Source code for Problem #7

Changed and new code marked in yellow highlight

```
#include <stdio.h>
#include "csim.h"

#define SIM_TIME 1.0e6

FACILITY Server1;
FACILITY Server2;
TABLE DelayTable;

void generate(double lambda, double mu1, double mu2);
void queue1(double service_time, double mu2, double clock);
void queue2(double service_time, double clock);

void sim(void)
{
  double   lambda;
  double   mu1, mu2;

  create("sim");

  Server1 = facility("Server #1");
  Server2 = facility("Server #2");
  DelayTable = table("Delay table");

  lambda = 1.0;
  mu1 = 3.0;
  mu2 = 2.0;

  generate(lambda, mu1, mu2);
  hold(SIM_TIME);

  // Output results
  printf("============================================================ \n");
  printf("==     *** CSIM M/M/1 queueing system simulation ***       == \n");
  printf("============================================================ \n");
  printf("= lambda   = %f cust/sec   \n", lambda);
  printf("= mu1      = %f cust/sec   \n", mu1);
  printf("= mu2      = %f cust/sec   \n", mu2);
  printf("============================================================ \n");
  printf("= Mean response time = %f sec      \n", table_mean(DelayTable));
  printf("============================================================ \n");
}

void generate(double lambda, double mu1, double mu2)
{
  double   interarrival_time;
  double   service_time;

  create("generate");

  while(1)
  {
    interarrival_time = exponential(1.0 / lambda);
    hold(interarrival_time);

    service_time = exponential(1.0 / mu1);
    queue1(service_time, mu2, clock);
  }
}

void queue1(double service_time, double mu2, double orgTime)
{
  create("queue1");

  reserve(Server1);
  hold(service_time);
  release(Server1);
```

```
  service_time = exponential(1.0 / mu2);
  queue2(service_time, orgTime);
}

void queue2(double service_time, double orgTime)
{
  create("queue2");

  reserve(Server2);
  hold(service_time);
  release(Server2);
  record((clock - orgTime), DelayTable);
}
```