ECE/CS 541: Computer System Analysis

Homework #2 (**Due 10/07/18**)

Student Name: Student's name *Netid: netid*

The intent of this homework is to get you some hands on experience with combinatorial modeling, specifically reliability block diagrams and fault trees. This homework will also serve as an introduction to Markov Chains. The problems will vary in difficulty from those that you should be able to solve in a couple of minutes, to those that require more thought.

Please note that this homework has a substantial coding and plotting component and thus we will use **digital submissions** through `compass2g`. For drawing block diagrams and fault trees, it is acceptable that you draw them on a piece of paper (clearly!) and then include a picture of that in your submission, though for your future and for our eyes, it is recommended that you use software packages for that. We highly recommend that you get acquainted with LaTeX's `TikZ` package since it can produce very professional-looking diagrams. All figures in this document, including the chess board, were generated using `TikZ`.

Submission Instructions

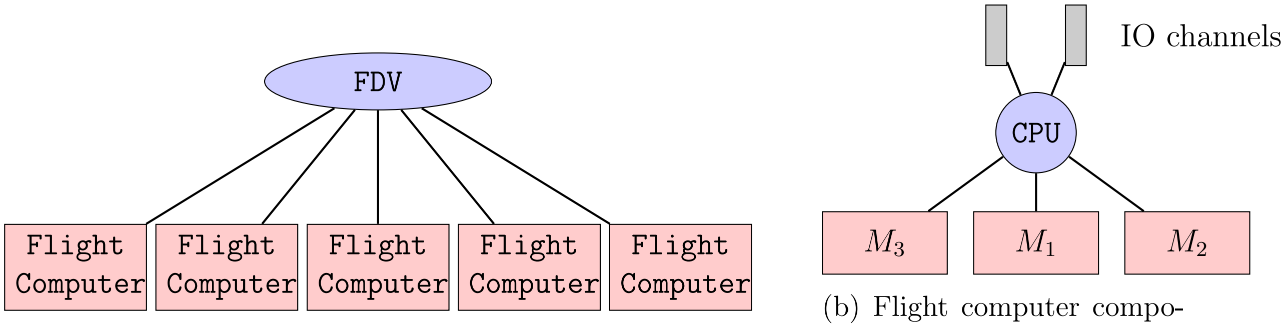
Submit a single zip file (or tarball) to `compass2g`, named `netid_hw2.zip(.tar.gz)` that contains the following:

- A **single PDF file** that contains all of your answers to the questions, along with plots and their analysis.
- The source code that you used for this submission. Please make sure to correctly label each file with the exercise it corresponds to. For example, if you use Python to plot the diagram for Exercise 1, then include that as `netid_ex1.py`. *There is no need to include the source code in your PDF document.*
- **You need to show ALL of the steps that you took in order to solve a problem and obtain a solution.** If you only state the answer as if it is a fact, you will get zero credit for that.

Exercise 1 (Combinatorial Modeling) (30 points)

The original space shuttle's flight control computer system had five identical flight control computers, operating on the same inputs. The outputs of these computers were compared to detect any failures. The overall control system required at least three of the five flight computers to be operational, to support "majority voting" in case of any failure.

Consider a similar system shown in Figure 1 where there is "fault-detector/voting" (FDV) component that merges and compares the outputs of the flight computers. The



(a) High level view of the FDV and its components.

(b) Flight computer component.

Figure 1: Space shuttle flight control system for Exercise 1.

overall system is operational if the FDV gets inputs from at least three operational flight control computers.

The connection between a flight control computer and an FDV is composed of *two independent* IO channels. The connection is operational as long as at least one of the channels is operational.

Each flight computer is composed of the following:

- a Central Processing Unit (CPU),
- a bank of three memory modules, M_1, M_2, and M_3,
- two identical IO channels that both connect to the FDV.

For a flight computer to be operational it is necessary that the CPU be able to read and write to at least two memory modules, and be connected to the "fault detector/voting component" by a least one IO channel.

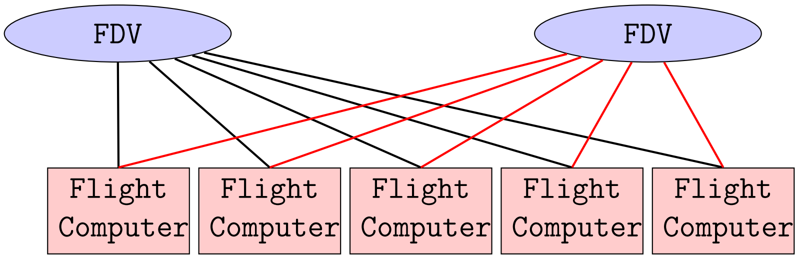
The failure distributions of each one of the constituent components are shown in Table 1. We assume that all failures occur independently.

Component	Failure Distribution
FDV	Exponential($\lambda = 10^{-7}$)
CPU	Weibull($\lambda = 10^{-7}, \beta = 2$)
M_1	Exponential($\lambda = 10^{-7}$)
M_2	Exponential($\lambda = 2 \times 10^{-7}$)
M_3	Exponential($\lambda = 3 \times 10^{-7}$)
IO channel	defined by a hazard rate $h(t) = t \times 10^{-10}$

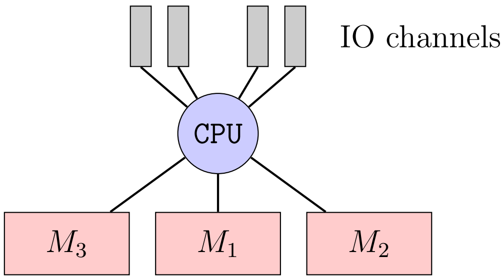
Table 1: Failure distributions for components in Exercise 1.

(a) Model the system using a Reliability Block Diagram.
(b) Express the failure modes of this system in terms of a fault tree.
(c) Build the Boolean formula that captures the failure conditions for this system.
(d) Use the programming and computational tools we introduced in Homework 0 (or others if you like) to perform the following:

- compute the mean time of failure (MTTF) of the system
- compute the probability that the system survives up to times $t_0 = 10000$, $t_1 = 100000$, and $t_2 = 1000000$.



(a) High level view of the FDV and its components.



(b) Flight computer component.

Figure 2: Space shuttle flight control system for Exercise 1(e).

- Plot the hazard rate function of the overall system, the cdf of the time to failure of the system, and the hazard rate function of a single flight computer's subsystem.

(e) A newly introduced reliability engineer has suggested that you modify the system as illustrated in Figure 2. The system now includes two more IO channels per flight control computer (Figure 2a) and two FDVs (Figure 2b). The system is operational if either FDV is operational and has at least three operational flight control computers connected to it.

Which of the two systems is more reliable? Why? How much is it more reliable? Explain your set up to answer the question and **support your argument with computations and plots**.

Exercise 2. (DTMC Warmup) (25 points)

Consider a model of a time-division multiplexed communication protocol with the following characteristics

1. There are N hosts connected to a shared communication medium (e.g., a bus).
2. Time is segmented in "slots", t_1, t_2, ...
3. Every host begins in the "offering load" state. When in this state, it waits to attempt its next communication a random number of slots that is geometrically distributed with success parameter p_s.
4. If two or more hosts attempt communication in the same time slot, we assume there is a collision and none of the communications succeed. The collision is detected, and every host that attempted a communication "backs off" and tries to communicate again after a number of time slots that is geometrically distributed with parameter p_r. A host that is waiting to retry a communication is in the "backed-off" state.
5. If a transmission is successful, it happens instantaneously, i.e., with time $t = 0$.

We consider the state of the network to be the number of hosts that are in the "offering load" state.

Build a DTMC of this communication network by answering the following questions:

1. Given that the DTMC is in state k at time t, under what conditions does it remain in state k at time $t + 1$? Write a formula for this transition probability.
2. Given that the DTMC is in state k at time t, under what conditions does it transition to state $k - 1$ at time $t + 1$? Write a formula for this transition probability.
3. Given that the DTMC is in state k at time t, under what conditions does it transition to state $j < k - 1$ at time $t + 1$? Write a formula for this transition probability.
4. Given that the DTMC is in state k at time t, under what conditions does it transition to state $k + j (j > 0)$ at time $t + 1$? Write a formula for this transition probability.
5. Give the general shape of the transition matrix P of this DTMC.

Assume now that $N = 3$.

1. Draw the DTMC and state its transition matrix P.
2. Does a stationary probability vector π exist for this DTMC? Why?
3. If π exists, give its formula as a function of p_r and p_s.

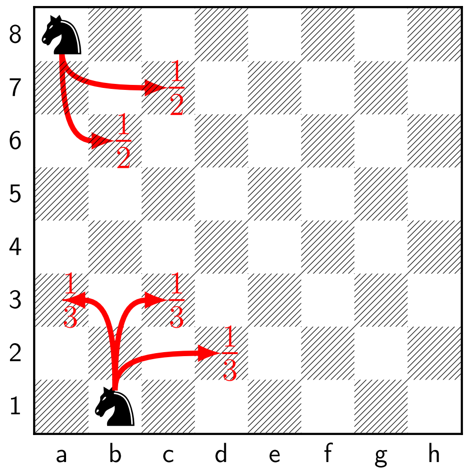


Figure 3: Chess board for exercise 4

Exercise 3. Fun with DTMCs and the laws of probability (20 points)

Let X be a discrete time Markov Chain taking values in $\{1, 2\}$ with a probability transition matrix

$$P = \begin{pmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{pmatrix}$$

Let Y be another random process defined as follows:

$$Y_k = \begin{cases} X_k & \text{with probability } 0.9 \\ X_k - 1 & \text{with probability } 0.1 \end{cases}$$

Find $\lim_{k\to\infty} P\left(X_k = 1 \mid Y_k = 1\right)$

Exercise 4. Let's play some chess (25 points)

Consider the chess board shown in Figure 3. Assume that there is only one piece (the black knight in this case) that has the standard possible moves for a knight in chess. For example, starting from position $b1$, the knight can move into $a3$, $c3$, or $d2$. A drunkard is attempting to play chess (alone, using only the black knight) and his strategy is the following. At every stage, move the knight to one of the possible move positions with equal probability. For example, from position $b1$, he will move the knight into either $a3$ or $c3$ or $d2$ each with probability $\frac{1}{3}$. Alternatively, starting from position $a8$, he can move the knight into position $b6$ or $c7$ each with probability $\frac{1}{2}$.

Assume that the knight is initially in position $b1$, what is the average number of steps taken by the player until the knight returns back to position $b1$? Explain how you set up and solved this problem **in detail** and include any source code you use for your solution. You can use any software packages you deem useful for your computations.
