COMP 9121 Assignment 1

You need to write your answers to all questions in a single .pdf file. Please name your file as Lastname\_Firstname\_assignment1.pdf. You may also submit your Python codes (questions 1 and 3). Please name your files as Lastname\_Firstname\_CRC.py and Lastname\_Firstname\_Dijkstra.py. Then you should zip your files in a single .zip package, using the name Lastname\_Firstname\_assignment1.zip. **The submission deadline is September 16, 5pm, Sydney time.**

There are four questions in this assignment. **You are required to answer two of them, but you are encouraged to answer all of them.** The full mark of each question is 10.

If you answer more than two questions, your 3rd and 4th highest marks of the questions will be halved, and **your total score is limited by 20**. For example, if you get 8,7,4,2 for the four questions, your final score will be min[8+7+4/2+2/2, 20]=18. If you get 8,7,6,8, your final score will be min[8+8+7/2+6/2, 20]=20.

If you are not comfortable with Python programming, you can skip questions 1 and 3. It is possible to get 20 if you only answer two questions.

Questions 1 and 2 are related to the link layer. You are suggested to work on these two questions as soon as possible. Questions 3 and 4 are related to the network layer. You are suggested to work on these two questions no earlier than week 5.

1. CRC Simulation

In this part, you are going to implement a simulator of cyclic redundancy check (CRC) code and test its performance. You can reuse parts of codes in Lab 2. The procedure is summarized as follows:

1. Randomly generate an N-bit information stream. You need to set N=8 this time. (You can reuse the code in Lab 2.)
2. Using the generator G=1001, generate the CRC bits and derive an (N+3)-bit coded stream. (Hint: implement long division in this step.)
3. Send the coded stream into a random flipping channel with bit-flip probability p. You need to test three cases: p=0.1, p=0.05, and p=0.02. (You can reuse the code in Lab 2.)
4. The receiver checks the received bits. (Hint: implement long division in this step). Then, there will be three possibilities:
	1. None of the bits are flipped.
	2. Some of the bits are flipped, and this is detected by the CRC.
	3. Some of the bits are flipped, but this is not detected by the CRC.
5. Repeat the above procedure many times (e.g., 100000 or more). Find out the probabilities of A), B), and C).

To self-check if you have done correctly, you can check with the following websites

Online CRC calculation

<https://www.ghsi.de/CRC/>

Binary to Decimal to Hexadecimal Converter

<https://www.mathsisfun.com/binary-decimal-hexadecimal-converter.html>

Questions

1. Use your simulator to find out the simulated probabilities of A), B), and C) under p=0.1, p=0.05, and p=0.02.
2. Compute the theoretical probabilities of A), B), and C), and compare them with your simulation results. (Hint: Please consider the error patterns divisible by 1001. Approximations can be made in this step, i.e., consider those “common” error patterns but ignore those “rare” error patterns.)

You need to submit your code in a separate file, using the name Lastname\_Firstname\_CRC.py.

2. Link Layer Performance

In the illustrated network, computers can only be located on either one of the two circles, C1 or C2. All computers on C1 are connected to hub H1, and all computers on C2 are connected to hub H2. No computer is located at the intersection of the two circles. The radius of each circle is 50 meters, and the maximum rate of all links is 1 Gbps. The propagation speed in the medium is 2×108 meters/second.



(1) Assume that the network uses Slotted Aloha, where the length of each timeslot is equal to the transmission time of a single packet. Each computer generates 500 packets per second with each packet being 1000 bytes. What is the throughput of the system if there are 10 computers on the network?

(2) In part (1), find the number of nodes in the system if about 30% of the timeslots is idle.

(3) In part (2), what is the expected delay for the successful transmission of each packet?

(4) What is the maximum number of nodes that can be supported in the network?

(5) What is the maximum number of nodes supported in the network if the Ethernet protocol is used?

3. Dijkstra’s Algorithm

In this question, you will implement Dijkstra’s algorithm in Python and find the shortest paths from one source node to any nodes in the network. There are n nodes in the network. The edge lengths (costs) are represented by an n\*n matrix. The i-th row and j-th column of the matrix indicates the length (or cost) from node i to node j. If there is no edge between node i and node j, their length (cost) is set to be 65535. In Python, we use a two-dimensional list to indicate the edge lengths (costs). For example, suppose n=3 (the indices of the nodes are 0, 1, and 2), then we set m=[[0,1,65535],[1,0,2],[65535,2,0]] to represent the following graph. The lengths from node 0 to node 0, 1, and 2 are 0, 1, and 65535, respectively. The lengths from node 1 to node 0, 1, and 2 are 1, 0, and 2, respectively. The lengths from node 2 to node 0, 1, and 2 are 65535, 2, and 0, respectively.



Note that the length from node i to itself is 0, and the length from node i to node j is equal to the length from node j to node i.

In the program, use Dijkstra’s algorithm to find the shortest distances from an arbitrary source node to any nodes. (You do not need to use a heap to find the minimum. It is acceptable to use min() to find the minimum in a list.) You must use m, n, and v to denote the two-dimensional matrix, number of nodes, and the index of source node respectively. m, n, and v must be assigned at the beginning (i.e., first three lines) of your codes. Your output should be in the following format [a0,a1,…,an-1], where a0,a1,…,an-1 denote the shortest distances from the source node (v) to node 0, node 1, …, node (n-1).

Example:

Here come your Python codes

|  |
| --- |
| m=[[0,1,65535],[1,0,2],[65535,2,0]]n=3v=0############## Your code## Your code## Your code## Your code## Your code############ |

Output: [0,1,3]

Reason: the shortest distances from node 0 (v) to nodes 0, 1, 2 are 0, 1, 3 respectively.

Warning: Your code will be tested against various test cases (under different m, n, and v). You need to submit your code in a separate file, using the name Lastname\_Firstname\_Dijkstra.py.

4. IP and Routing

(1) In the following figure, fill out the forwarding table at router A. Choose the shortest path in this problem. The cost is the hop count (note, each network, labelled by cloud, is one hop).



Routing table at Router A

|  |  |  |  |
| --- | --- | --- | --- |
| **Destination Address** | **Cost** | **Interface** | **Next hop IP address** |
| 128.4.100.0/24 | 2 | M1 | 128.5.101.10 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

(2) Is it possible to aggregate these addresses to have a smaller routing table? If so, to what? If not, why not?

(3) Now assume that the whole network 128.4.100.0/24 breaks down, and assume that the routers are using Routing Information Protocol (RIP) with Poison Reverse. How many cycles of routing table updates does it take for A to find out that the network 128.4.100.0/24 is unreachable? Show all the steps in the following table. Show each entry as the pair (next router, cost).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| iterations | Router A | Router B | Router C | Router D |
| Before the break | ( B , 2 ) | ( — , 1 ) | ( D , 2 ) | ( — , 1 ) |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |
| … |  |  |  |  |

(4) If RIP without Poison Reverse is used, how many cycles does it take for A to learn that the network 128.4.100.0/24 is unreachable? Show all the steps in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| iterations | Router A | Router B | Router C | Router D |
| Before the break |  |  |  |  |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |
| … |  |  |  |  |