Shahzodjon Ismatov

110518374

ESE346

Project # 1

***Brief Description of the Program***

I utilized Java to complete this project. The program asks the user for the number of timeslots to be generated, and uses that to generate packets per time slot by using the Bernoulli Process method defined. This method compares the p value to a random value seeded with current time, and if p is less than the generated value, it increments succeed. If succeed is 3 or greater, the value of passed packets is set to 3. Otherwise, they’re set to 0,1, or 2 based on the increments of succeed variable. The method then returns the passed packets, which are then summed and divided by the number of timeslots in order to get the average number of busy and dropped packets.

Then the user is prompted to save the data. I used the BufferedWriter class to write to a file which the user can define by simply entering the file name followed by the extension of the file(such as .csv, .xlsx, etc). This makes it easy to use Microsoft Excel for data manipulation, such as graphing to see the graph distribution of the data.

**Source Code**

/\*\*

\* **@author** Shahzod Ismatov

\* SBU ID: 110518374

\* ESE 346 Project 1

\* Switching Element simulator

\*/

**import** java.io.BufferedWriter;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.util.Random;

**import** java.util.Scanner;

**public** **class** **SwitchingElement** {

**public** **static** **void** **main**(**String**[] args) **throws** **IOException** {

**Random** **r** = **new** Random(**System**.*currentTimeMillis*());

**Scanner** **s** = **new** Scanner(**System**.***in***);

**System**.***out***.print("Enter the number of time slots: ");

**int** **timeSlots** = **Integer**.*parseInt*(s.nextLine());

**System**.***out***.print("Enter file save name: ");

**String** **fileName** = s.nextLine();

**BufferedWriter** **bw** = **new** BufferedWriter(**new** FileWriter(fileName));

//adding the header

bw.append("p");

bw.append(",");

bw.append("Avg Outputs");

bw.append(",");

bw.append("Avg Dropped");

**int**[] **outputs** = **new** **int**[timeSlots]; //an array to store outputs

**double** **sumPass** = 0, **sumDrop** = 0;

**double** **avgOutputs**[] = **new** **double**[timeSlots]; //an array to store the average number of outputs;

**double** **avgDropped**[] = **new** **double**[timeSlots]; //an array to store the avg dropped packets

**for** (**double** **p** = 0.05; p < 1.0; p += 0.05) { //for each probability p

**int** **j** = 0;

**for** (**int** **i** = 0; i < timeSlots; i++) { //for each timeSlot at a given probability

**int** **genPacket** = r.nextInt(10) + 1; //generate a random packet [1-10] inputs

outputs[i] = *BernoulliProcess*(p, genPacket); //Run the bernoulli process for that p value

sumPass += outputs[i]; //calculate the total passes

sumDrop += (genPacket - outputs[i]); //calculate the total drops

}

//Calculate the average dropped / passed packets

avgOutputs[j] = sumPass / timeSlots;

avgDropped[j] = sumDrop / timeSlots;

**System**.***out***.println("Avg Output: " + avgOutputs[j]);

**System**.***out***.println("Avg Dropped: " + avgDropped[j]);

*saveToFile*(bw, avgOutputs, avgDropped, p, j, fileName);

j++;

sumPass = 0;

sumDrop = 0;

}

bw.close();

s.close();

}

**public** **static** **int** **BernoulliProcess**(**double** p, **int** genPacket) {

**int** **succeed** = 0, **passed** = 0, **dropped** = 0;

**for** (**int** **i** = 0; i < genPacket; i++) { //for each generated packet at a given timeslot

**if** (*uniform*() < p) //randomly generates a probability, compares to p, and passes the packet if true

succeed++;

}

**if** (succeed > 3) //if the number of packets succeeded is 3 or greater

passed = 3; //output gets 3 (max # of outputs)

**else** //otherwise

passed = succeed; //outputs gets 0, 1, 2

dropped = genPacket - passed;

**System**.***out***.format("\nProbability: %.2f \n", p);

**System**.***out***.println("Generated Packets: " + genPacket);

**System**.***out***.println("Passed: " + passed);

**System**.***out***.println("Dropped: " + dropped + "\n");

**return** passed;

}

//save to a specifed file based on extension (e.g. .csv , .xls, etc)

**public** **static** **void** **saveToFile**(**BufferedWriter** bw, **double** avgOutputs[], **double** avgDropped[], **double** p, **int** j,

**String** fileName) **throws** **IOException** {

**try** {

**for** (**int** **i** = 0; i <= j; i++) {

bw.append("\n");

bw.append(**String**.*valueOf*(p));

bw.append(",");

bw.append(**String**.*valueOf*(avgOutputs[i]));

bw.append(",");

bw.append(**String**.*valueOf*(avgDropped[i]));

}

bw.flush();

//System.out.println("Save successful!\n");

} **catch** (**Exception** **e**) {

**System**.***out***.println(e.getMessage());

}

}

/\*\*

\*Generates a random real number uniformly in [0, 1).

\*/

**public** **static** **double** **uniform**() {

**Random** **r** = **new** Random();

**return** r.nextDouble();

***Flow Chart***

Enter timeslots

v

Save to File

Calculate

Avg. # busy packets

Avg. # dropped packets

Bernoulli Process

Is

p < 1.0?

No

Yes

p = p + 0.05

Is

i < slots?

No

i = i + 1

Yes

Generate packets

**Conclusion**

This project made me really think about how a switching element functions as a whole. The probability of a packet arrival at a given input modeled through a Bernoulli distribution, in combination with the redirection to output, conditions for output and discrete time simulation were the key concepts in this project.

The graphs for *average number of busy slots vs p* as well as *average number of dropped packets vs p* are shown. The former has an increasing curve which later stabilizes to almost flat, while the latter has an upward curve like an exponential function. This makes since the higher the probability of success, the more likely to get an average number 3 packets, whereas the lower the probability of success, the more likely to have dropped packets. I did a super test for a case of 100,000 timeslots and the results were as expected. The number of average busy packets increased as the probability *p* increased, and the number of average dropped packets decreased.