Министерство образования Республики Беларусь

Учреждение высшего образования

«Белорусский государственный университет

информатики  и радиоэлектроники»

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

Лабораторная работа № 1 по дисциплине

ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

на тему

**«РЕШЕНИЕ ЗАДАЧ МОДЕЛИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ ИМИТАЦИИ СЛУЧАЙНЫХ ВЕЛИЧИН НА ОСНОВЕ МЕТОДА МОНТЕ-КАРЛО»**

Вариант 1

Выполнили студент группы 720601:

Агинский В.Д.

Пелецкий А.Е.

Свило С.А.

Проверил:

Cевернев А.М.

Минск 2020

## 1 Цель работы

1. Изучить алгоритмы имитации случайных величин на основе метода Монте-Карло и методы проверки этих алгоритмов.

2. Разработать алгоритмы имитации случайных величин, необходимых для решения задачи моделирования согласно варианту задания.

## 2 Постановка задачи

В банк данных, состоящий из двух баз данных (БД1 и БД2), поступают запросы. Время обработки поступившего запроса – экспоненциальная случайная величина со средним значением 15 мс. Затем выполняется поиск запрашиваемой информации в БД1, а если она там не обнаруживается – то в БД2. Информация в БД1 обнаруживается примерно для 60% запросов; для остальных запросов она обнаруживается в БД2. Время поиска информации в каждой из баз данных – случайная величина, распределенная по экспоненциальному закону со средним значением 30 мс. Примерно в 30% случаев объём ответа составляет ровно 300 символов, в 10% – рóвно 6000 символов. В остальных случаях объём ответа – случайная величина (V); на основании наблюдений установлено, что плотность распределения этой величины может быть приближённо задана графиком на рисунке 1.

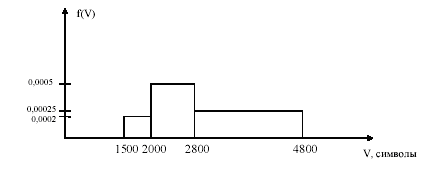


Рисунок 1

Скорость формирования ответа – 6 мс на каждые 80 символов.

Требуется определить:

– среднее время ответа;

– вероятность того, что время ответа превысит 100 мс.

## 3 Ход выполнения

Разработаем алгоритмы имитации случайных величин, необходимых для решения задачи моделирования согласно полученному варианту задания.

Первая величина – время обработки поступившего запроса. Как видно из условия, данная величина является экспоненциальной случайная величиной со средним значением 15 мс. Для имитации экспоненциальной случайной величины здесь и далее будем использовать данную формулу:

.

Ниже приведена реализация функции, имитирующая случайную величину времени обработки поступившего запроса, реализованной на языке *JavaScript*.

function expon(average) {

const r = Math.random();

return -average \* Math.log(r);

}

function processingTime(average) {

return expon(average);

}

То же самое выполним для времени поиска запрашиваемой информации. Данная случайная величина также распределена по экспоненциальному закону, однако также известна разная вероятность обнаружения запрашиваемой информации в разных базах данных. Учтем этот момент, реализация функции, имитирующей данную случайную величину приведена ниже.

function databaseSearch(average) {

let totalSearchTime = expon(average);

if (Math.random() > 0.6) {

totalSearchTime += expon(average);

}

return totalSearchTime;

}

Следующая величина – объём ответа. Для данный величины имеется график плотности распределения. Это значит, что для получения функции имитации данной случайной величины необходимо воспользоваться методом обратных функций.

По графику, приведённому на рисунке 1, запишем уравнение плотности распределения времени внесения добавок:

Функция распределения времени внесения добавок имеет следующий вид:

Из уравнений находим выражение , т.е. формулу (алгоритм) для имитации случайной величины :

Таким образом, реализация функции, имитирующая случайную величину объема ответа, а также скорости его формирования представлены ниже.

function getNumberOfSymbols() {

const r1 = Math.random();

if (r1 < 0.3) {

return 300;

}

if (r1 >= 0.3 && r1 < 0.9) {

const r2 = Math.random();

if (r2 < 0.1) {

return (r2 + 0.3) / 0.0002;

} else if (r2 >= 0.1 && r2 < 0.5) {

return (r2 + 0.9) / 0.0005;

}

return (r2 + 0.2) / 0.00025;

}

return 6000;

}

function timeToFormResponse() {

const chunk = 80;

return 6 \* Math.ceil(getNumberOfSymbols() / chunk);

}

Теперь необходимо посчитать общие затраты времени, определить среднее время ответа и вероятность того, что время ответа превысит 100 мс. Часть кода, реализующая данные вопросы представлена ниже.

function totalTimeCost() {

return processingTime(15) + databaseSearch(30) + timeToFormResponse();

}

function app() {

const N = 100000;

let timeBeforeHundred = 0;

let sumTime = 0;

for (let i = 0; i < N; i++) {

const time = totalTimeCost();

sumTime += totalTimeCost();

if (time < 100) {

++timeBeforeHundred;

}

}

const averageTime = sumTime / N;

console.log(`Average time: ${averageTime}`);

console.log(`Probability: ${timeBeforeHundred / N}`);

}

Проверим алгоритм имитации случайных величин с помощью гистограммы.

Результаты выполнения программы приведены в таблице 1.

Таблица 1

|  |  |
| --- | --- |
| Интервал | Относительная частота |
| 1500-1800 | 0,00008158 |
| 1800-2100 | 0,00011053 |
| 2100-2400 | 0,00027368 |
| 2400-2700 | 0,00027105 |
| 2700-3000 | 0,00014211 |
| 3000-3300 | 0,00014737 |
| 3300-3600 | 0,00009737 |
| 3600-3900 | 0,00011053 |
| 3900-4200 | 0,00012632 |
| 4200-4500 | 0,00011053 |
| 4500-4800 | 0,00013421 |

На рисунке 2 приведена гистограмма, построенная по значениям относительных частот. Видно, что гистограмма близка к графику плотности распределения имитируемой величины, приведенному на рисунке 1. Таким образом, можно предполагать, что алгоритм имитации правильный.

Рисунок 2 – Гистограмма

Проверим алгоритм имитации на основе критерия «хи-квадрат».

Теоретические вероятности попадания случайной величины в каждый из интервалов представлены в таблице 2.

Таблица 2

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Интервал | 1500 1800 | 1800 2100 | 2100 2400 | 2400 2700 | 2700 3000 | 3000 3300 | 3300 3600 | 3600 3900 | 3900 4200 | 4200 4500 |
| Вероятность, | 0.06 | 0.09 | 0.15 | 0.1 | 0.075 | 0.075 | 0.075 | 0.075 | 0.075 | 0.075 |

Найдем значение критерия «хи-квадрат»:

Найдем квантиль распределения “хи-квадрат” . Зададим уровень значимости α=0,05. Число степеней свободы определяется следующим образом: s=k-1=11-1=10. В данном примере . Так как ., можно утверждать, что случайные величины соответствуют плотности распределения.

Выполним те же два метода проверки для времени обработки запроса. Приведём гистограмму для времени обработки сигнала со средним значением 15 мс (рис. 3).

Рисунок 3 – Гистограмма для времени обработки сигнала

Проверим алгоритм имитации на основе критерия «хи-квадрат». Найдем его значение:

Найдем квантиль распределения “хи-квадрат” . Зададим уровень значимости α=0,05. Число степеней: s=k-1=16-1=15. В данном примере . Так как ., можно утверждать, что случайные величины соответствуют плотности распределения.

## ВЫВОДЫ

Таким образом, в ходе выполнения данной лабораторной работы мы изучили алгоритмы имитации случайных величин на основе метода Монте-Карло и методы проверки этих алгоритмов. Разработали собственные алгоритмы имитации случайных величин, необходимых для решения задачи моделирования согласно полученному варианту задания. Составили моделирующий алгоритм, вычисляющий средние затраты времени, а также вероятность того, что оно превысит 100 мс. Проверили алгоритмы имитации для двух величин: заданной графиком плотности распределения и заданной экспоненциальным законом. Для проверки воспользовались изученными методами: на основе гистограммы и по критерию «хи-квадрат». Убедились в их правильности.