*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| Gerb-BMSTU_01 | ***«Московский государственный технический университет  имени Н.Э. Баумана***  ***(национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

**Отчет**

**по лабораторной работе № 1**

**Дисциплина: Моделирование**

**Название лабораторной работы: Марковские процессы**

Студент гр. ИУ6-53 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**Мамадаев И.М.\_\_\_

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_**Шайхутдинов А.А.\_

(Подпись, дата) (И.О. Фамилия)

Москва, 2018

**Задание:**

Написать программу моделирующую марковские процессы с дискретным и непрерывным временем. Для каждого случая построить 1000 шагов. Найти финальные вероятности попадания в то или иное состояние процесса эмпирческим и аналитическим путем. Входные данные марковских процессов, как число состояний, начальный вектор, матрицы переходов заполнить самостоятельно.

**Марковская цепь с дискретным временем**

**import** UIKit

**class** DynamicViewController: UIViewController, UITableViewDataSource, UITableViewDelegate

**@IBOutlet** **weak** **var** table: UITableView

**var** mas2: [String] = []

**var** StartStatus = [0.5, 0.2, 0.3]

**var** Matrix = [[0.6,0.2,0.2],

[0.3,0.3,0.4],

[0.1,0.7,0.2]]

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

table.dataSource = **self**

table.delegate = **self**

disc\_time()

table.reloadData()

}

**func** tableView(**\_** tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

**var** cell = tableView.dequeueReusableCell(withIdentifier: "mas2")

**if** cell == **nil** {

cell = UITableViewCell(style: .default, reuseIdentifier: "mas2")

}

cell?.textLabel?.text = mas2[indexPath.row]

**return** cell!

}

**func** tableView(**\_** tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

**return** mas2.count

}

**func** disc\_time() {

**var** curstatus = 0

**var** st1 = 0.0

**var** st2 = 0.0

**var** st3 = 0.0

**var** random = 0.0

**for** **\_** **in** 0..<100 {

**var** r1 = Int(arc4random())

**var** r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

**if** random > (StartStatus[0]+StartStatus[1]) {

curstatus = 2

} **else** {

**if** random < StartStatus[0]{

curstatus = 0

} **else** {

curstatus = 1

}

}

st1 = 0; st2 = 0; st3 = 0

**for** i **in** 0...1000 {

r1 = Int(arc4random())

r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

**if** random < Matrix[curstatus][0] {

curstatus = 0

st1 += 1

} **else** {

**if** random > (Matrix[curstatus][2]+Matrix[curstatus][1]+Matrix[curstatus][0]){

curstatus = 2

st3 += 1

} **else** {

curstatus = 1

st2 += 1

}

}

mas2.append(String(i)+" Step, status "+String(curstatus))

}

}

mas2.append("Emperic P1 = "+String(st1/1000)+", "+String(StartStatus[0]\*Matrix[0][0]+StartStatus[1]\*Matrix[1][0]+StartStatus[2]\*Matrix[2][0]))

mas2.append("Emperic P2 = "+String(st2/1000)+", "+String(StartStatus[0]\*Matrix[0][1]+StartStatus[1]\*Matrix[1][1]+StartStatus[2]\*Matrix[2][1]))

mas2.append("Emperic P3 = "+String(st3/1000)+", "+String(StartStatus[0]\*Matrix[0][2]+StartStatus[1]\*Matrix[1][2]+StartStatus[2]\*Matrix[2][2]))

mas2.reverse()

table.reloadData()

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Марковская цепь с непрерывным временем**

**import** UIKit

**class** StaticViewController: UIViewController, UITableViewDataSource, UITableViewDelegate {

**@IBOutlet** **weak** **var** table: UITableView!

**var** mas1: [String] = []

**var** StartStatus = [0.5, 0.2, 0.3]

**var** Matrix = [[0.6,0.2,0.2],

[0.3,0.3,0.4],

[0.1,0.7,0.2]]

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

table.dataSource = **self**

table.delegate = **self**

const\_time()

table.reloadData()

}

**func** tableView(**\_** tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

**var** cell = tableView.dequeueReusableCell(withIdentifier: "mas1")

**if** cell == **nil** {

cell = UITableViewCell(style: .default, reuseIdentifier: "mas1")

}

cell?.textLabel?.text = mas1[indexPath.row]

**return** cell!

}

**func** tableView(**\_** tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

**return** mas1.count

}

**func** const\_time() {

**var** curstatus = 0

**var** st1 = 0.0

**var** st2 = 0.0

**var** st3 = 0.0

**var** ts1 = 0.0

**var** ts2 = 0.0

**var** ts3 = 0.0

**var** random = 0.0

**for** **\_** **in** 0..<100 {

**var** r1 = Int(arc4random())

**var** r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

**if** random < StartStatus[0]{

curstatus = 0

} **else** {

**if** random > (StartStatus[0] + StartStatus[1]){

curstatus = 2

} **else** {

curstatus = 1

}

}

mas1.append("Step, status "+String(curstatus))

**for** i **in** 0...1000 {

r1 = Int(arc4random())

r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

ts1 = -log(random) / Matrix[curstatus][0]

r1 = Int(arc4random())

r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

ts2 = -log(random) / Matrix[curstatus][1]

r1 = Int(arc4random())

r\_max = INT\_MAX

random = Double(r1) / Double(r\_max)

ts3 = -log(random) / Matrix[curstatus][2]

**if** (ts1 < ts2)&&(ts1 < ts3){

curstatus = 0

st1 += ts1

} **else** {

**if** (ts2 < ts1)&&(ts2 < ts3){

curstatus = 1

st2 += ts2

} **else** {

curstatus = 2

st3 += ts3

}

}

mas1.append(String(i)+" Step, status "+String(curstatus))

}

}

mas1.append("P1 = "+String(st1 / (st1 + st2 + st3)))

mas1.append("P2 = "+String(st2 / (st1 + st2 + st3)))

mas1.append("P2 = "+String(st2 / (st1 + st2 + st3)))

mas1.reverse()

table.reloadData()

}

}

**Результат работы программы**

****

Рисунок 1 – главное меню программы

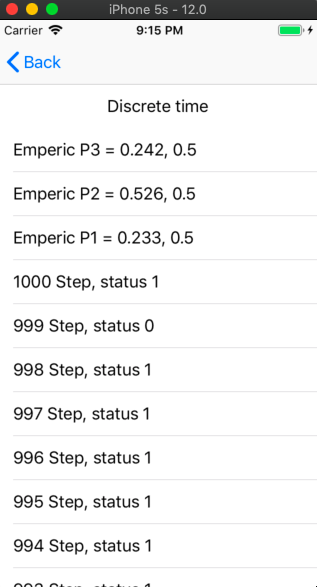
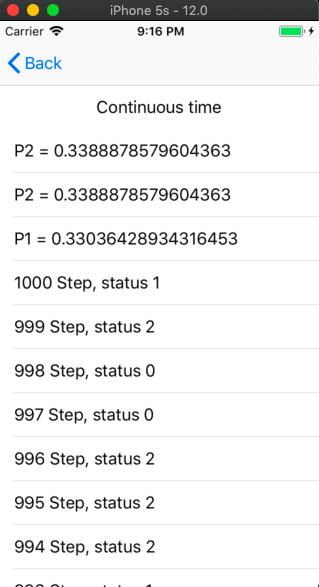
**** ****

Рисунок 2 – Дискретное время Рисунок 3 – Непрерывное время

**Вывод**

В результате выполнения данной лабораторной работы было написано iOS-приложение на языке Swift для моделирования работы марковских цепей с дискретным и непрерывным временем. Программа строит таблицы с 1000 шагами и выводит эмпирически и аналитически полученные вероятности нахождения в состояниях. Полученные значения получаются различными ввиду ограниченности количества шагов и зависимости от случайно сгенерированных величин.