ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ. ЗАНЯТИЕ №3

ТЕОРИЯ

1. Пусть S – множество состояний некоторой системы, T – множество моментов времени, в каждый из которых система может либо поменять своё состояние, либо остаться в том же состоянии, в котором была.

2. В науке и на практике рассматриваются случаи, когда множество состояний системы S либо дискретное (конечное или счётное), либо непрерывное (равномощно множеству вещественных чисел $\mathbb{R}$; множества большей мощности рассматривать не будем). Также дискретным или непрервным может быть время T.

3. Далее мы рассмотрим ситуацию, когда оба множества S и T *дискретны*. Пусть, вдобавок, множество S конечно и имеет n элементов.

4. Положим, что переход из одного состояния в другое *случаен*.

5. Введём обозначения. Пусть $\mathsf{P}(t, j)$ – вероятность того, что в момент времени t система находится в состоянии с номером j ($j = 1, 2, \ldots, n$); $\mathsf{P}(t, i, j)$ – вероятность того, что в момент времени t система перешла из состояния с номером i в состояние с номером j (это так называемая *переходная вероятность*, $i, j = 1, 2, \ldots, n$). В начальный момент времени $t = 0$ система также с некоторой вероятностью могла находиться в каком-то состоянии из S, следовательно, нужно задать вектор начальных вероятностей $\mathsf{P}(0, j)$ ($j = 1, 2, \ldots, n$).

6. Ясно, что $\mathsf{P}(0, 1) + \mathsf{P}(0, 2) + \cdots + \mathsf{P}(0, n) = 1$ (в начальный момент времени мы имеем дискретное распределение вероятностей того, что система будет в состояниях с номерами $1, 2, \ldots, n$).

7. Аналогично, $\mathsf{P}(t, 1) + \mathsf{P}(t, 2) + \cdots + \mathsf{P}(t, n) = 1$ (в момент времени t мы имеем дискретное распределение вероятностей того, что система будет в состояниях с номерами $1, 2, \ldots, n$).

8. Наконец, для любого $i = 1, 2, \ldots, n$ выполняется $\mathsf{P}(t, i, 1) + \mathsf{P}(t, i, 2) + \cdots + \mathsf{P}(t, i, n) = 1$ (в момент времени t мы имеем дискретное распределение вероятностей того, что система перейдёт из состояния с номером i в состояния с номерами $1, 2, \ldots, n$).

   Так как S конечно, то можно определить для каждого $t \in T$ матрицу переходных вероятностей:

$$\mathsf{P}(t) = \begin{pmatrix} \mathsf{P}(t,1,1) & \mathsf{P}(t,1,2) & \ldots & \mathsf{P}(t,1,n) \\ \mathsf{P}(t,2,1) & \mathsf{P}(t,2,2) & \ldots & \mathsf{P}(t,2,n) \\ \ldots & \ldots & \ldots & \ldots \\ \mathsf{P}(t,n,1) & \mathsf{P}(t,n,2) & \ldots & \mathsf{P}(t,n,n) \end{pmatrix}$$

9. Если вероятности $\mathsf{P}(t,k)$ однозначно определяются тем, в каких состояниях система была в моменты времени $0,1,\dots,t-1$, то описываемый процесс называется *цепью Маркова*.

10. Если вероятность $\mathsf{P}(t,k)$ зависит только от того состояния, в котором система находилась в момент времени $t-1$, и не зависит от того, в каких состояниях система находилась в моменты времени $0,1,\dots,t-2$, то цепь Маркова называется *одиородой*. Иначе говоря, в однородной цепи Маркова переходные вероятности не зависят от времени: $\mathsf{P}(t,i,j)=\mathsf{P}(i,j)$. Имеем матрицу переходных вероятностей:

$$\mathsf{P}=\begin{pmatrix} \mathsf{P}(1,1) & \mathsf{P}(1,2) & \dots & \mathsf{P}(1,n) \\ \mathsf{P}(2,1) & \mathsf{P}(2,2) & \dots & \mathsf{P}(2,n) \\ \dots & \dots & \dots & \dots \\ \mathsf{P}(n,1) & \mathsf{P}(n,2) & \dots & \mathsf{P}(n,n) \end{pmatrix}$$

11. Используя формулу полной вероятности, получаем для неоднородной Марковской цепи:

$$\mathsf{P}(t,j)=\mathsf{P}(t-1,1)\cdot\mathsf{P}(t-1,1,j)+\mathsf{P}(t-1,2)\cdot\mathsf{P}(t-1,2,j)+\dots+$$
$$+\mathsf{P}(t-1,n)\cdot\mathsf{P}(t-1,n,j).$$

Это равносильно тому, что

$$(\mathsf{P}(t,1),\dots,\mathsf{P}(t,n))=(\mathsf{P}(t-1,1),\dots,\mathsf{P}(t-1,n))\cdot\mathsf{P}(t-1)$$

Чтобы получить вектор распределения вероятностей состояний на шаге с номером t, необходимо вектор распределения вероятностей состояний на шаге с номером $t-1$ умножить справа на матрицу переходных вероятностей $\mathsf{P}(t-1)$.

12. По индукции получаем:

$$(\mathsf{P}(t,1),\dots,\mathsf{P}(t,n))=(\mathsf{P}(0,1),\dots,\mathsf{P}(0,n))\cdot\mathsf{P}(0)\cdot\mathsf{P}(1)\cdot\ldots\cdot\mathsf{P}(t-1).$$

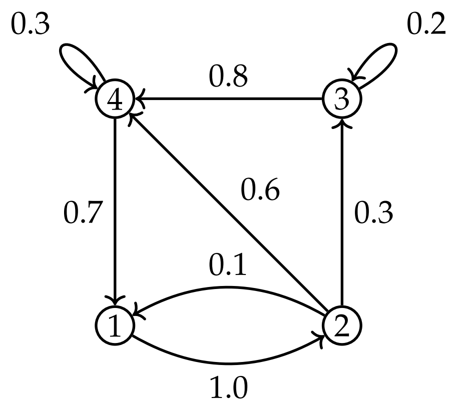
13. Для случая однородной цепи:

$$(\mathsf{P}(t,1),\dots,\mathsf{P}(t,n))=(\mathsf{P}(0,1),\dots,\mathsf{P}(0,n))\cdot\mathsf{P}^t.$$

14. Для случая *одиородиой* цепи имеет место теорема: если на каком-то шаге t_0 матрица P^{t_0} состоит из строго положительных чисел, то существуют пределы, к которым стремятся вероятности $\mathsf{P}(t,j)$:

$$\mathsf{P}(\infty,j)=\lim_{t\to\infty}\mathsf{P}(t,j)\quad(j=1,2,\dots,n).$$

15. Цепь Маркова можно визуально представить в виде ориентированного взвешенного графа (возможно, с петлями), в котором вершины – состояния системы, дуги – переходы от одного состояния к другому, а веса дуг – вероятности перехода. Заметим, что для каждой вершины сумма весов дуг, исходящих из этой вершины, должна быть равна единице.



16. Где появляются марковские процессы?

    - Процесс посещаемости сайтов (состояние – web-ресурс, переход между состояниями – переход по ссылке). С помощью марковских моделей можно выполнить ранжирование web-ресурсов.
    - Системы массового обслуживания (состояние – число заявок, находящиихся в системе в данный момент, переход между состояниями – изменение количества заявок в системе).
    - Социально-экономические системы. Например, если какая-то социальная группа делится на подгруппы, то можно в качестве состояний системы взять вектор количества членов в каждой подгруппе. Переход между состояниями – изменение объёма членов в подгруппах. Так, указанный подход может быть применён при изучении клиентской базы какой-либо организации (банк, вузы, магазины и пр.). Можно использовать данный подход при изучении миграции населения (состояние – вектор численностей населения в странах мира, переход между состояниями – изменение этих количеств в результате миграции и процессов рождения и смертности).
    - Если состояния – это транспортные узлы, а переход между состояниями – это транспортировка какого-либо объекта из одного узла в другой, то функционирование транспортного графа также является марковской цепью. Такого рода модель может применяться при изучении транспортировки грузов, почтовых отправлений, людей.
    - Моделирование генератора текстов. Для того, чтобы автоматически сгенерировать некоторый текст, нужно иметь лингвистический корпус – набор реальных текстов, из которых вычисляется вероятность составления пар слов. Отсюда, слова – это состояния системы, а переход между состояниями – составление словосочетания из двух слов.
    - Технические системы. Состояние системы – это состояние какого-то технического объекта («работает», «сломан», «в ремонте», «списан»), а переход между состояниями – это переход из одного технического состояния в другое (например, из состояния «сломан» в состояние «списан»).

17. Следует различать процесс получения вероятностных распределений на состояниях системы от имитации работы системы. Имитация работы за-

ключается в том, что в каждый момент времени мы генерируем какое-то *определённое* состояние, используя вероятностное распределение на множестве состояний, тем самым получая некоторую *фактическую траекторию перемещения между состояниями*.

18. Итак, мы имеем многошаговый процесс. Обозначим номер состояния системы в момент времени t через $i(t)$. Перед началом итераций нужно задать начальное распределение вероятностей состояний и сгенерировать, пользуясь этим распределением, номер начального состояния системы $i(0)$. Далее, на каждой итерации с номером t необходимо выбирать в матрице переходных вероятностей строку с номером $i(t-1)$ и расценивая эту строку как распределение вероятностей нового состояния, сгенерировать номер состояния $i(t)$.

19. Код программы.

```
from random import randint, random
from matplotlib import pyplot


# функция, возвращающая неотрицательный вектор,
# сумма элементов которого равна единице -
# вектор вероятностного распределения
def dist_generate(numb):
    container = [randint(0, 100) for _ in range(numb)]
    s = sum(container)
    container = [elem / s for elem in container]
    return container


# функция, возвращающая случайный номер состояния
# по заданному распределению состояний
def random_state(prob_vector):
    r = random()
    s = prob_vector[0]
    numb = len(prob_vector)
    for k in range(numb):
        if r < s:
            return k
        else:
            s += prob_vector[k + 1]
    return None


# произведение вектора на матрицу
def vec_mul_matr(vec, mat):
    result = []
    for j in range(len(vec)):
        sumx = 0
        for k in range(len(vec)):
            sumx += vec[k] * mat[k][j]
        result.append(sumx)
    return result
```

```python
# общее количество состояний
n = 30
# максимальное количество шагов
t_max = 100
# начальное распределение вероятностей состояний
# помещаем в контейнер, сохраняющий распределения
p_vector = dist_generate(n)
# контейнер состояний
s_vector = []
# матрица переходных вероятностей
p_matrix = [dist_generate(n) for _ in range(n)]
# основной цикл
time = []
for t in range(t_max):
    time.append(t)
    s_vector.append(random_state(p_vector))
    p_vector = vec_mul_matr(p_vector, p_matrix)
# вывод графика состояний
pyplot.plot(time, s_vector)
pyplot.title("График состояний системы")
pyplot.show()
# вывод результатов
stat_list = [[k, s_vector.count(k)] for k in range(n)]
print(stat_list)
```

Результаты:


```
[[0, 3], [1, 5], [2, 6], [3, 4], [4, 5], [5, 3], [6, 2],
 [7, 4], [8, 7], [9, 3], [10, 6], [11, 2], [12, 1], [13, 3],
 [14, 3], [15, 3], [16, 4], [17, 3], [18, 2], [19, 6], [20, 1],
 [21, 1], [22, 3], [23, 1], [24, 4], [25, 5], [26, 3], [27, 2],
 [28, 1], [29, 4]]
```

ПРАКТИКА

Составить программу, моделирующую марковский процесс (на любом удобном вам языке).

Критерии оценивания:

1. Воспроизвёл программу из лекции (просто набрал python-скрипт) и поэкспериментировал в ней: 0 баллов.
2. За основу взята программа из лекции, но сделаны изменения в модель: 2 балла
3. Составил программу самостоятельно на языке, отличном от языка программы из лекции, снабдив программу минимальными возможностями (организован процесс перехода из одного состояния в другое и как результат выводится частота нахождения в каждом состоянии): 4 балла
4. Добавлено графическое сопровождение (показ переходов между состояниями): 6 баллов.
5. В модель добавлены дополнительные возможности (имеется поглощающее состояние, модель прогоняется несколько раз и вычисляется среднее время до попадания в поглощающее состояние, либо показано, что распределение вероятностей нахождения в том или ином состоянии приближается к предельному): 8 баллов.