МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра телекоммуникационных систем и вычислительных средств
(ТС и ВС)

Отчет по лабораторной работе №4
по дисциплине
Теория массового обслуживания

по теме:
ЦЕПИ МАРКОВА И СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ

Студент:
Группа ИА-331 *Я.А Гмыря*

Предподаватель:
Преподаватель *А.В Андреев*

Новосибирск 2025 г.

СОДЕРЖАНИЕ

ЦЕЛЬ И ЗАДАЧИ

Цель: Изучение методов создания и анализа цепей Маркова

Задание к лабораторной работе

Задание к лабораторной работе

1. Создайте .m файл в MATLAB и сгенерируйте в нем матрицу переходов P в соответствии с вариантом в таблице 4.1.

2. Создайте цепь Маркова на основе полученной матрицы переходов, используя функцию ‘dtmc()’, задав названия состояний

29

«Healthy», «Unwell», «Sick», «Very sick». Присвойте ее переменной MC.

3. Выведите в консоль матрицу переходов полученной цепи, используя функцию ‘MC.P’. Обратите внимание, что полученная матрица является нормированной. Используя функцию ‘sum()’, убедитесь, что все строки матрицы дают в сумме 1. Сохраните полученную матрицу для отчета.

4. Постройте граф матрицы при помощи функции ‘graphplot()’. Используя аргументы данной функции, покажите вероятности переходов различным цветом. Сохраните полученное изображение для отчета.

5. Используя нормированную матрицу, постройте кумулятивную матрицу переходов, в которой каждое значения в последующем столбце матрицы являются суммой предыдущих. Назовите эту матрицу P_cum.

6. Промоделируйте поведение цепи Маркова в течение 200 итераций, используя следующее выражение:

$$z_{t+1} = \sum_k \left(r > P_{cum(z_t,k)}\right) + 1,$$

где r – случайное число, распределенное равномерно на интервале [0,1]; P_cum – кумулятивная матрица переходов, z_t – состояние цепи в момент времени t.

В качестве начального состояния цепи, укажите состояние 1.

Рисунок 1 — Задание для лабораторной работы

7. Используя функцию 'plot()', постройте график, показывающий, как в течение 200 наблюдений менялось состояние, в котором находилась цепь. Сохраните этот график для отчета.

8. Повторите пункт 6 для 1000 и 10000 итераций.

9. Рассчитайте оценку цепи Маркова по полученным наблюдениям для 200, 1000 и 10000 итераций, используя следующее выражение:

$$P_{obs}\ (z_t, z_{t+1}) = \sum_k P_{obs}\ (z_t, z_{t+1}) + 1$$

И затем нормализовав полученные матрицы переходов. Сохраните их для отчета. Повторите пункты 3 и 4 для каждой из полученных матриц.

10. Сравните результаты, полученные в пунктах 4 и 9 (одна цепь Маркова, полученная в пункте 3, и еще три цепи, полученные в пункте 9), сделайте выводы. Разместите матрицы переходов и графы для этих четырех цепей рядом в отчете.

11. Повторите пункты 6 и 7 для цепи Маркова, полученной для 200 наблюдений в пункте 8. Разместите эти графики рядом в отчете. Сравните полученные графики. Сделайте выводы.

12. Системы массового обслуживания. Рассмотрите СМО в

30

соответствии с вашим вариантом. Напишите код для расчета всех показателей эффективности вашей СМО. Сведите все показатели в таблицу и добавьте ее к отчету вместе с характеристиками системы. Является ли данная система эффективной? Интерпретируйте результаты.

Рисунок 2 — Задание для лабораторной работы

Дополнительные задания

1. Создайте цепь Маркова при помощи языков программирования R, Python.

2. Постройте систему Массового обслуживания при помощи элементов MATLAB Simulink.

Рисунок 3 — Задание для лабораторной работы

Таблица 4.1. Коэффициенты в матрице переходных вероятностей

Вариант	p_{11}	p_{12}	p_{13}	p_{14}	p_{21}	p_{22}	p_{23}	p_{24}	p_{31}	p_{32}	p_{33}	p_{34}	p_{41}	p_{42}	p_{43}	p_{44}
1	0.8	0.1	0.1	0	0.2	0.7	0.1	0	0.1	0.2	0.5	0.2	0.2	0.2	0.2	0.4
2	2	5	3	4	8	5	4	2	5	4	3	2	1	5	4	4
3	13	14	0	0	1	23	34	1	3	3	3	3	2	1	15	1
4	0	3	3	3	3	0	2	1	3	2	0	1	3	2	1	0
5	15	1	1	1	1	15	1	1	1	1	15	1	1	1	1	15
6	5	5	0	0	0	5	5	0	0	3	3	0	0	0	3	3
7	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1
8	5	5	1	0	5	5	2	0	2	5	5	2	0	0	3	3
9	50	10	1	1	50	20	2	2	20	50	50	10	10	10	50	50
10	1	0.1	0	0	0.1	1	0.1	0	0	0.1	1	0	0	0	0.2	1

Таблица 4.2. Характеристики системы массового обслуживания

Вариант	λ	μ	n	m
1	5	1	1	5
2	10	5	2	10
3	15	10	3	5
4	20	20	4	10
5	25	30	5	5
6	30	40	6	10
7	35	50	7	5
8	40	30	8	10
9	45	20	9	5
10	50	10	10	10

31

Рисунок 4 — Варианты заданий

ТЕОРИЯ

Теория:

Основные сведения

Основы теории Марковских цепей явились исходной базой общей теории случайных процессов, а также таких важных прикладных наук, как теория диффузионных процессов, теория надежности, теория массового обслуживания и т.д. В настоящее время теория Марковских процессов и ее приложения широко применяются в самых различных областях.

Благодаря сравнительной простоте и наглядности математического аппарата, высокой достоверности и точности получаемых решений, особое внимание Марковские процессы приобрели у специалистов, занимающихся исследованием операций и теорией принятия оптимальных решений.

Примечание. Для выполнения лабораторной работы необходимы базовые знания о цепях Маркова. Необходимо знать понятия: матрица переходов, конечные состояния, граф.

Цепи Маркова

Цепь Маркова – последовательность случайных событий с конечным или счётным числом исходов, характеризующаяся тем свойством, что, говоря нестрого, при фиксированном настоящем будущее независимо от прошлого.

Матрица переходов

Переходы в цепях Маркова могут быть заданы при помощи матрицы переходов, в которой каждый элемент матрицы p_{ij} показывает вероятность перехода цепи из состояния i в состояние j.

$$P = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N1} & \cdots & p_{NN} \end{pmatrix}$$

Системы массового обслуживания

Система массового обслуживания (СМО) – система, которая производит обслуживание поступающих в нее требований. Обслуживание требований в СМО осуществляется обслуживающими приборами. Классическая СМО содержит от одного до бесконечного числа приборов.

Многоканальная СМО с ограниченной длиной очереди. Имеется n каналов, на которые поступает поток заявок с интенсивностью λ. Поток

28

Рисунок 5 — Теория для лабораторной работы обслуживании каждого канала имеет интенсивность μ. Длина очереди – m. Найти предельные вероятности состояний системы и показатели ее эффективности.

Показатели эффективности СМО

Ключевые показатели:

– Абсолютная пропускная способность системы (А) – среднее число заявок, обслуживаемых в единицу времени;

– Относительная пропускная способность (Q) – средняя доля поступивших заявок, обслуживаемых системой;

– Вероятность отказа (Ротк) – вероятность того, что заявка покинет СМО не обслуженной.

Другие показатели:

– Среднее количество занятых каналов (kзан);

– Среднее количество заявок в системе (Lсист);

– Среднее время пребывания заявки в системе (Тсист);

– Средняя длина очереди (Lоч);

– Среднее время ожидания заявки в очереди (Точ).

Для многоканальной СМО с ограниченной длиной очереди эти характеристики рассчитываются следующим образом:

$$P_{\text{отк}} = p_{n+m} = \frac{\rho^{n+m}}{n^m n!} p_0$$

$$Q = 1 - P_{\text{отк}}$$

$$A = \lambda Q$$

$$\bar{k}_{\text{зан}} = \frac{A}{\mu} = \rho Q$$

$$L_{\text{оч}} = \frac{\rho^{n+1}}{nn!} \cdot \frac{1 - \left(\frac{\rho}{n}\right)^m \left(m + 1 - \frac{m}{n}\rho\right)}{\left(1 - \frac{\rho}{n}\right)^2} \rho_0$$

$$T_{\text{оч}} = \frac{L_{\text{оч}}}{\lambda}$$

$$L_{\text{сист}} = L_{\text{оч}} + \bar{k}_{\text{зан}}$$

$$T_{\text{сист}} = \frac{L_{\text{сист}}}{\lambda}$$

Рисунок 6 — Теория для лабораторной работы

ХОД РАБОТЫ

Цепь маркова

Зададим матрицу переходов для цепи Маркова в соответствии с вариантом. Создадим цепь Маркова с помощью встроенной функции matlab и выведем нормированную матрицу переходов. Мы нормируем матрицу, чтобы она отражала вероятности перехода между состояниями, т.к матрица, данная в задании, отражает только то, как эти вероятности соотносятся. С помощью встроенной функции выведем визуализацию цепи Маркова в виде графа.

```
Normalized matrix:
    0.5000    0.5000         0         0
    0.2941    0.2941    0.1765    0.2353
    0.5000    0.5000         0         0
    0.3333    0.3333    0.2222    0.1111
```

Рисунок 7 — Нормированная матрица переходов

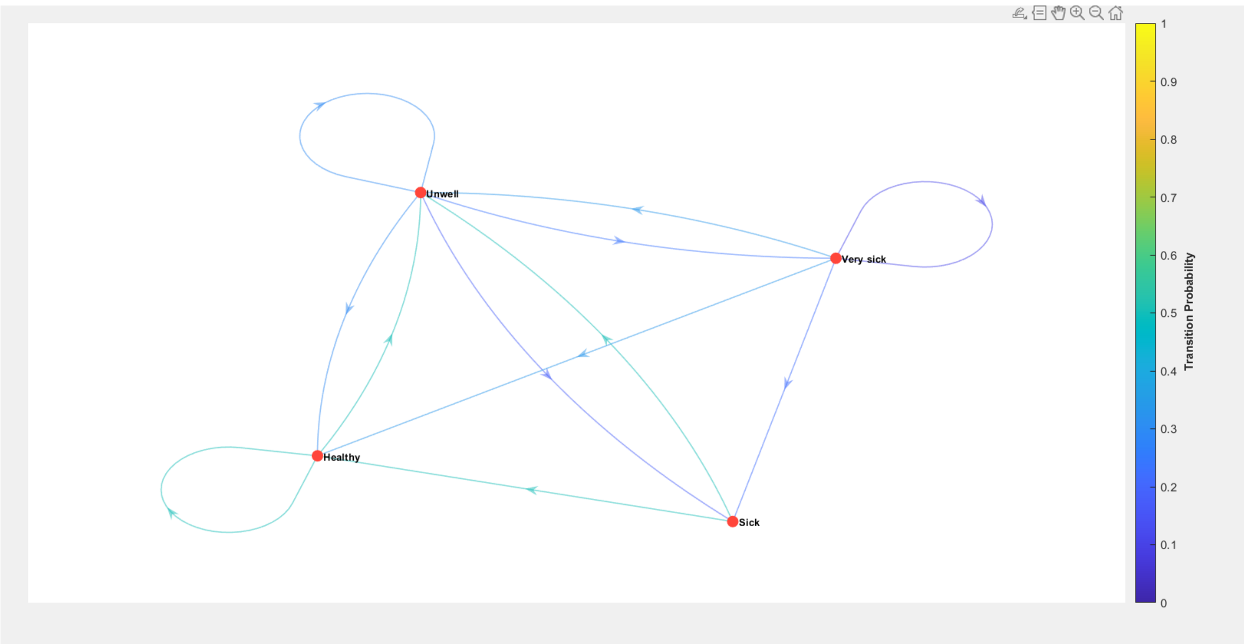

Рисунок 8 — Представление цепи Маркова в виде графа

Данный граф отражает, сколько состояний у системы, как состояния связаны между собой и с какой вероятностью могут переходить в другие состояния.

С помощью встроенной функции созадим кумулятивную матрицу - матрицу, в каждой строке которой каждое значение является суммой всех предыдущих. Если сумма становится > 1, то значение округляется до 1.

```
Cumulative matrix
    0.5000    1.0000    1.0000    1.0000
    0.2941    0.5882    0.7647    1.0000
    0.5000    1.0000    1.0000    1.0000
    0.3333    0.6667    0.8889    1.0000
```

Рисунок 9 — Кумулятивная матрица

Эта матрица нам нужна будет в дальнейшем для моделирования переходов в цепи Маркова

Реализуем алгоритм, моделирующий переходы в цепи Маркова:

```
%markov chain modeling
figure;
for m = 1:length(N)

    % allocate memory
    z = zeros(N(m), 1);

    %define start value
    z(1) = 1;

    %generate vector of numbers from uniformity distribution
    r = rand(N(m), 1);

    for i = 2:N(m)
        k = 1;
        %compute new state
        while r(i-1) > P_cum(z(i-1), k)
            k = k + 1;
        end
        %save state
        z(i) = k;
    end

    %build plot
    subplot(3, 1, m)
    plot(z)

```

Рисунок 10 — Алгоритм моделировани цепи Маркова

В данном алгоритме сначала задается начальное значение z(1) = 1. Это показывает, в каком состоянии (вершине графа) находится система на начало моделирования. Далее будем сравнивать вероятность перехода в другую вершину (пробегаемся по строке) с числом из отрезка [0;1] сгенерированного по законам нормального расрпеделения. Это делается потому, что марковский процесс случаен, т.е нет правила переходов между состояниями, есть только вероятность, с которой это можно сделать. Сравнение со случайным числом как раз вносит эту самую случайность при моделировании. Вычисляем следующую вершину и сохраняем для последующих вычислений.

Результаты моделирования для N = 200, 1000, 10000. N - кол-во временных отсчетов или кол-во переходов:

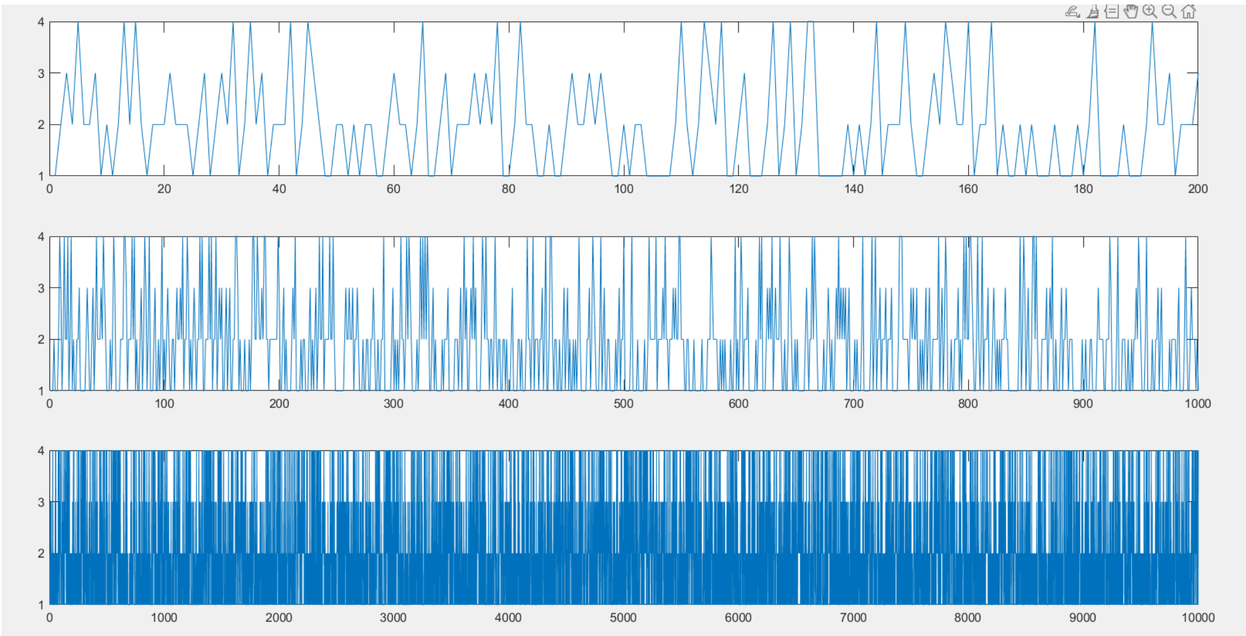

Рисунок 11 — Результаты моделирования

На графиках по оси Y отложены состояния системы, а по оси X - временные отсчеты. График иллюстрирует переходы в цепи Маркова.

Вычислим оценку цепи Маркова:

```
    %allocate memory
    P_obs = zeros(4,4);

    %compute estimate for markov chain
    for n = 2:N(m)
        prev = z(n-1);
        curr = z(n);
        P_obs(prev, curr) = P_obs(prev, curr) + 1;
    end

    %normalazing matrix
    for row = 1: size(P_obs, 2)
        s = 0;
        for col = 1: size(P_obs, 1)
            s = s + P_obs(row, col);
        end

        for col = 1: size(P_obs, 1)
            P_obs(row, col) = P_obs(row, col) / s;
        end
    end

    %final matrix
    disp(P_obs)

end
```

Рисунок 12 — Алгоритм оценки цепи Маркова

Данный алгоритм по сути вычисляет, сколько раз было сделано переходов в то или иное состояние системы, а потом нормализует полученную матрицу.

Результат:

```
N = 200
    0.6024    0.3976         0         0
    0.2692    0.3462    0.1667    0.2179
    0.4118    0.5882         0         0
    0.1905    0.3810    0.2381    0.1905

N = 1000
    0.5091    0.4909         0         0
    0.2799    0.3445    0.1627    0.2129
    0.5376    0.4624         0         0
    0.2039    0.4175    0.2427    0.1359

N = 10000
    0.5074    0.4926         0         0
    0.3025    0.2862    0.1679    0.2434
    0.5091    0.4909         0         0
    0.3666    0.3162    0.2080    0.1091
```

Рисунок 13 — Оценка цепи Маркова

Можем заметить, что данная матрица очень похожа на исходную матрицу переходов, причем с увеличением N матрица становится всё более похожей на исходную. Это говорит о том, что система, построенная по законам цепи Маркова со временем сохраняет свои начальные характеристики (матрицу переходов).

Расчет характеристик СМО

Реализация:

```
%define params
lambda = 30;
mu = 40;
n = 6;
m = 10;

%compute mark chain params
rho = lambda/mu;

p0 = 1 / (sum(arrayfun(@(k) rho^k/factorial(k), 0:n-1)) + rho^n/factorial(n) * (1-rho^m)/(1-rho));

P_otk = rho^(n+m)/(n^m*factorial(n)) * p0;

Q = 1 - P_otk;

A = lambda * Q;

k_zan = A / mu;

L_och = (rho^(n+1)/(n*factorial(n))) * (1 - (rho/n)^m * (m+1 - m*rho/n)) / (1 - rho/n)^2 * p0;

T_och = L_och / lambda;

L_sist = L_och + k_zan;

T_sist = L_sist / lambda;

%output result
disp(['rho = ', num2str(rho)]);
disp(['P_otk = ', num2str(P_otk)]);
disp(['Q = ', num2str(Q)]);
disp(['A = ', num2str(A)]);
disp(['k_zan = ', num2str(k_zan)]);
disp(['L_och = ', num2str(L_och)]);
disp(['T_och = ', num2str(T_och)]);
disp(['L_sist = ', num2str(L_sist)]);
disp(['T_sist = ', num2str(T_sist)]);
```

Рисунок 14 — Расчет параметров СМО

Результат:

```
>> IA_331_lab4_2
rho = 0.75
P_otk = 1.0871e-13
Q = 1
A = 30
k_zan = 0.75
L_och = 1.9058e-05
T_och = 6.3526e-07
L_sist = 0.75002
T_sist = 0.025001
```

Рисунок 15 — Результат расчетов

Как результаты характеризуют такую СМО?

Такая СМО имеет большую пропускную способность (A) относительно среднего кол-ва заявок (Q), из это следует малые среднее количество занятых каналов (kзан), среднее количество заявок в системе (Lсист), среднее время пребывания заявки в системе (Tсист), средняя длина очереди (Lоч), среднее время ожидания заявки в очереди (Tоч) и малая вероятность отказа.

Пример СМО в simulink

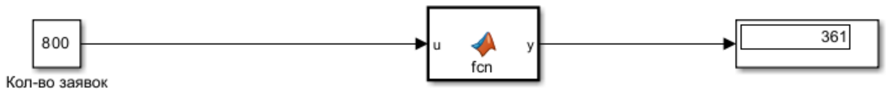

Рисунок 16 — Пример системы массового обслуживания

Блок с константой - число заявок, поданых на СМО. Саму логику СМО реализует функция-блок и кол-во обслуженных заявок передает на display.

Реализация функции-блока:

```
function y = fcn(u)
  %define params
   lambda = 30;
   mu = 20;
   n = 1;
   m = 5;

   %compute mark chain params
   rho = lambda/mu;

   p0 = 1 / (sum(arrayfun(@(k) rho^k/factorial(k), 0:n-1)) + rho^n/factorial(n) * (1-rho^m)/(1-rho));

   P_otk = rho^(n+m)/(n^m*factorial(n)) * p0;

   Q = 1 - P_otk;

   A = lambda * Q;

   k_zan = A / mu;

   L_och = (rho^(n+1)/(n*factorial(n))) * (1 - (rho/n)^m * (m+1 - m*rho/n)) / (1 - rho/n)^2 * p0;

   T_och = L_och / lambda;

   L_sist = L_och + k_zan;

   T_sist = L_sist / lambda;

   result = zeros(1000, 1);

   for i = 1 : u
      r = rand();

       if r >= P_otk
           result(i) = 1;
       else
           result(i) = 0;
       end
   end

   y = sum(result);
end
```

Рисунок 17 — Реализация СМО

В функции задаются начальные параметры, из которых высчитываются остальные. Самый важный параметр - вероятность отказа. Функция с вероятностью P либо обслуживает клиента, либо нет. Можно поиграться с параметрами и смотреть, как будет вести себя система. Можно сделать логику сложнее, чтобы участвовал не один параметр, а несколько.

Увеличим число каналов работы СМО и посмотрим, как это отразится на результате:

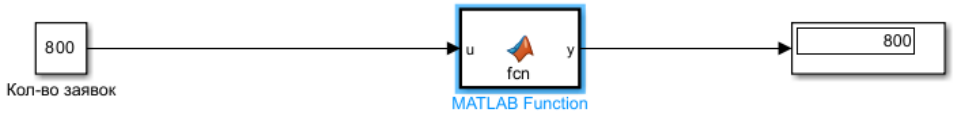

Рисунок 18 — Зависимость работы СМО от кол-ва каналов

Видим, что при увеличении числа каналов до 6 (до этого был всего один) вероятность обработать заявку кратно выросла.

ВЫВОД

В ходе работы я изучил свойства цепи Маркова путем моделирования переходов на языке matlab. Познакомился с параметрами СМО и рассчитал их.