МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»
КАФЕДРА №51

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

МЕТОД УКРУПНЕНИЯ МАРКОВСКОЙ ЦЕПИ ДЛЯ АНАЛИЗА АЛГОРИТМА ALOHA

по курсу: ТЕОРИЯ МНОЖЕСТВЕННОГО ДОСТУПА

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР.	5711М		Пятаков В.С.
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2018

Цель работы

Провести анализ характеристик алгоритма ALOHA с использованием аппарата Марковских цепей.

Задание

1. Выполнить имитационное моделирование алгоритма ALOHA;
2. Найти среднее количество заявок в очереди и среднюю задержку сообщения в системе;
3. Найти оценку среднего количества заявок в очереди и средней задержки с использованием Марковских цепей;
4. Построить график зависимости среднего количества заявок в очереди от интенсивности входного потока по результатам имитационного моделирования и оценки с использованием Марковских цепей;
5. Построить график зависимости средней задержки сообщения в системе от интенсивности входного потока по результатам имитационного моделирования и оценки с использованием Марковских цепей;
6. Сделать выводы по полученным результатам.

Порядок выполнения работы

Случайный множественный доступ

Алгоритм случайного доступа, рассматриваемый в лабораторной работе, относится к децентрализованным системам управления доступа к среде. При децентрализованном доступе устройства равноправны и по некоторому алгоритму они организовывают доступ к общему каналу. Предполагается синхронная система, то есть все абоненты имеют засинхронизированные часы (единая служба времени). Абоненты используют систему общего времени для доступа к каналу. Случайный доступ относиться к динамическому закреплению ресурса канала. Абонент передает сообщение по каналу сообщение готовое к передаче, если по каналу передаются сразу несколько сообщений от разных абонентов, то возникает конфликт, который разрешается по некоторому алгоритму.

Введем ряд допущений для рассматриваемой модели случайного множественного доступа:

1. Предполагается, что все сообщения у всех абонентов имеют одинаковую длину, время передачи одного сообщения принято за единицу времени. Все время передачи по каналу разбито на окна, длительность окна соответствует времени передачи одного

сообщения. Абоненты точно знают моменты разделения и могут начать передачу только в начале окна.

2. В окне возможно 3 события:
   2.1. Событие «Конфликт». В окне одновременно передают два абонента или больше. Считается, что из-за наложения сигналов сообщения полностью искажаются и не могут быть приняты правильно.
   2.2. Событие «Успех». В окне передает один абонент, в этом случае считается, что абонент успешно передает сообщение.
   2.3. Событие «Пусто». В окне никто не передает.
3. Абоненты наблюдают выход канала в конце окна и достоверно определяют какое из трех событий произошло.
4. В системе имеется M абонентов, в каждом окне у абонента с вероятностью y может появиться новое сообщение и с вероятностью $1-y$ сообщение не появляется. В среднем у всех абонентов в одну единицу времени возникает λ сообщений (интенсивность входного потока). Интенсивность входного потока у всех абонентов в системе одинакова и у каждого абонента она равна λ/M.

Алгоритмом случайного множественного доступа называется правило в соответствии, с которым каждый абонент, имеющий готовое к передаче сообщение, в начале каждого окна решает, передавать сообщение или нет. Существуют различные виды алгоритмов случайного множественного доступа, в данной лабораторной работе рассматриваются следующий алгоритм:

Алгоритм вероятностная ALOHA. Абоненты, имеющие к передаче сообщение, всегда передают его в канал с вероятностью p.

Моделирование пуассоновского входного потока

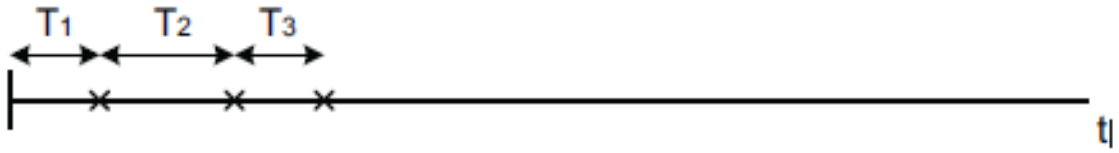

Рис. 1 - Пуассоновский входной поток

T_1 – интервал времени от 0 и до первого сообщения.

Будем считать, что T_i независимые и одинаково распределенные по экспоненциальному закону с интенсивностью λ случайные величины.

Используя равномерное распределение в интервале $U \in (0,1)$и метод обратной функции, можно получить случайные, распределенные по экспоненциальному закону величины T_i следующим образом:

$$T_i = \frac{\ln(U_i)}{-\lambda},$$

где $U_i \in (0,1)$.

X_i– количество заявок, которые возникли на интервале длины 1. Тогда, так как T_i одинаково распределенные по экспоненциальному закону и независимые случайные величины, то X_i будут независимыми одинаково распределенными случайными величинами и будут распределены по закону Пуассона.

$$\Pr\{X = k\} = \frac{\lambda^k}{k!} e^{-\lambda}$$

Если с течением времени очередь возрастает, то система называется неустойчивой. Для большинства систем интенсивность входного потока должна быть строго меньше, чем интенсивность обслуживания ($\lambda<\mu$).

Теорема Литтла: $d = \frac{\overline{N}}{\lambda}$, где $\overline{N}$ - среднее число заявок в системе.

Моделирование алгоритма ALOHA

С использованием методов имитационного моделироавния можно найти необходимые характеристики исследуемого алгоритма.

Для алгоритма ALOHA входными параметрами являются:

1. количество абонентов в системе;
2. вероятность передачи каждого абонента;
3. буфер для хранения сообщений для каждого абонента;
4. интенсивность входного потока заявок в систему.

В рассматриваемой модели вероятности передачи для всех абонентов и интенсивность входного потока равны, т.е. $p_1 = p_2 = ... = p_M$ и $\lambda_1 = \lambda_2 = ... = \lambda_M$.

Моделироавание представляет из себя 2 цикла, внешний по интенсивности входного потока λ и внутренний по окнам T.

Алгоритм работы

1. В начале окна проверяется не появилось ли новых заявок у абонентов:
   1.1. если появились, то смотрим имеются у него в буфере уже заявки:
      1.1.1. если буфер пустой кладем туда заявку и запоминаем окно,когда она пришла;
      1.1.2. если заполнен, то отбрасываем.
   1.2. если не появилось, то переходим к следующему шагу.
2. Далее смотрим наличие заявок в буфере у абонентов:
   2.1. если буфер не пустой:
      2.1.1. генерируем равномерно распределенное случайное число RAND и сравниваем его с вероятностью передачи абонента $p_i = \frac{1}{M}$
         2.1.1.1. Если p > RAND, то абонент может передать сообщение;
         2.1.1.2. Если p < RAND, то абонент не может передать сообщение.
   2.2. если буфер пустой, то переходим к следущему шагу.
3. Смотрим у скольких абонентов есть возможность передавать
   3.1. если у 2 и более абонентов ,то происходит событие КОНФЛИКТ, и в этом окне никто передать сообещение не сможет;
   3.2. если только у одного абонента , то он передает сообщение и происходит событие УСПЕХ. После передачи у него стирается из буфера сообщение.
   3.3. если 0, то переходим к следущему шагу.
4. В конце окна происходит следующее:
   4.1. если был абонент, который передал сообщение считается задержка этого сообщения в системе , как $delay_i = t_{input} - t_{output}$, прибавляется счетчик вышедших из системы сообщений ;
   4.2. если не был, то переходим к следущему шагу.
5. Считаем сколько заявок на момент конца окна в буфере N.

В результате моделирования, мы можем найти среднее количество заявок в буфере и среднюю задержку сообщения в системе поделим полученные величины на количество окон $mean\_delay = delay / T$, $mean\_N = N / T$.

Метод укрупнения состояний Марковской цепи

Рассмотрим систему с следующими параметрами:

1. число абонентов в системе $M = 2$;
2. буфер для хранения сообщения у каждого абонента b=1;
3. интенсивность входного потока $\lambda_1 = \lambda_2 = ... = \lambda_M$;
4. вероятность передачи сообщения $p_1 = p_2 = ... = p_M$.

Данную систему можно описать Марковской цепью с 4 состояниями, как показано на рисунке 2.

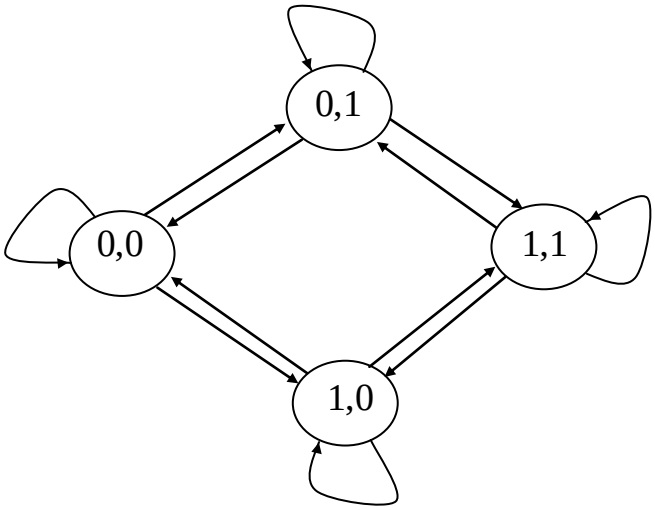

Рис. 2 - Марковская цепь с 4 состояниями

Данную систему можно описать Марковской цепью с 3 состояниями, где под состоянием, будет подразумеваться число абонентов, у которых есть сообщение, так как вероятности передачи у всех абонентов одинаковые и $P(0,1) = P(1,0)$.

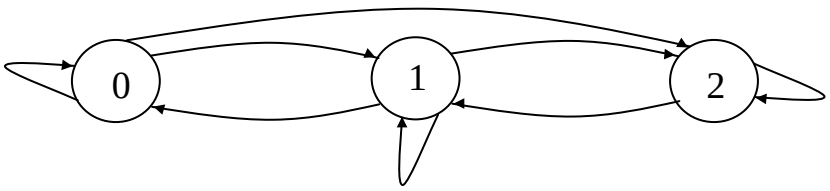

Рис. 3 - Марковская цепь с укруплением состояний для M=2

Посчитаем вероятности переходов из состояние в состояние:

1. $\Pr\{0 \mid 0\} = y^2$;
2. $\Pr\{1 \mid 0\} = y \cdot (1-y) + (1-y) \cdot y = 2 \cdot y \cdot (1-y)$;
3. $\Pr\{2 \mid 0\} = \Pr\{\text{появилось у} \geq 2 \text{ сообщение}\}$,

где y – сообщение у абонента не появилось, p – абонент передает сообщение.

Обобщим данный пример на M абонентов показанный на рисунке 3. В полученной Марковской цепи будет M+1 состояние

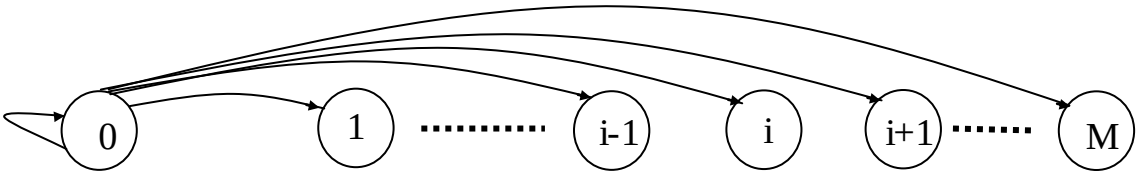

Рис. 4 - Марковская цепь с укруплением состояний для M абонентов

Опишем переходные вероятности данной Марковской цепи:

1. $\Pr\{0 \mid 0\} = y^M$
2. $\Pr\{i \mid 0\} = C_M^i y^{M-i} (1-y)^i$
3. $\Pr\{M \mid 0\} = (1-y)^M$
4. $\Pr\{i-1 \mid i\} = ip(1-p)^{i-1} y^{(M-i)}$
5. $\Pr\{i+j \mid i\} = C_{M-i}^j (1-y)^j y^{M-i-j} (1-ip(1-p)^{i-1}) + C_{M-i}^{j+1} (1-y)^{j+1} y^{M-i-(j+1)} ip(1-p)^{i-1}$
6. $\Pr\{M-1 \mid M\} = Mp(1-p)^{M-1}$
7. $\Pr\{M \mid M\} = 1 - Mp(1-p)^{M-1}$

Зная все вероятности переходов, можно составить матрицу переходов P. Используя матрицу переходов P, можно составить систему линейных уравнений и решив её, получить стационарное распределение Марковской цепи. После получения стационарного распределения можно найти теоретическую оценку среднего количества заявок в очереди N_theory и используя её, найти среднюю задержку сообщения в системе D_theory.

Опишем в общем виде алгоритм нахождения стационарного распределения Марковской цепи:

1. получить матрицу переходов P используя вероятности переходов Марковской цепи;
2. вычесть из нее транспонированной единичную матрицу $P^T - E = C$;
3. последнюю строку полученной матрицы C заполняем единицами

$$C = \begin{bmatrix} ..... \\ ..... \\ 1111 \end{bmatrix} = A;$$

4. вектор b заполняем нулями, последнее значение меняем на единицу

$$\overline{b} = \begin{bmatrix} 0 \\ ... \\ 1 \end{bmatrix};$$

5. находим стационарное распределение Марковской цепи

$$A\overline{p} = \overline{b}$$

$$A^{-1}A\overline{p} = A^{-1}\overline{b}$$

$$\overline{p} = A^{-1}\overline{b}$$

Получили вектор стационарного распределения $\overline{p} = (P(0), P(1), ..., P(M))$, с помощью его можем найти среднее число заявок в буфере, как:

$$\overline{N} = M[N] = \sum_{i=0}^{M} ip(i)$$

Так как в суммирование не участвует первый член, то вычитая его из единицы , можем получить выходную интенсивность.

$$\overline{D} = \frac{\overline{N}}{1 - P(0)}.$$

Результаты моделирования

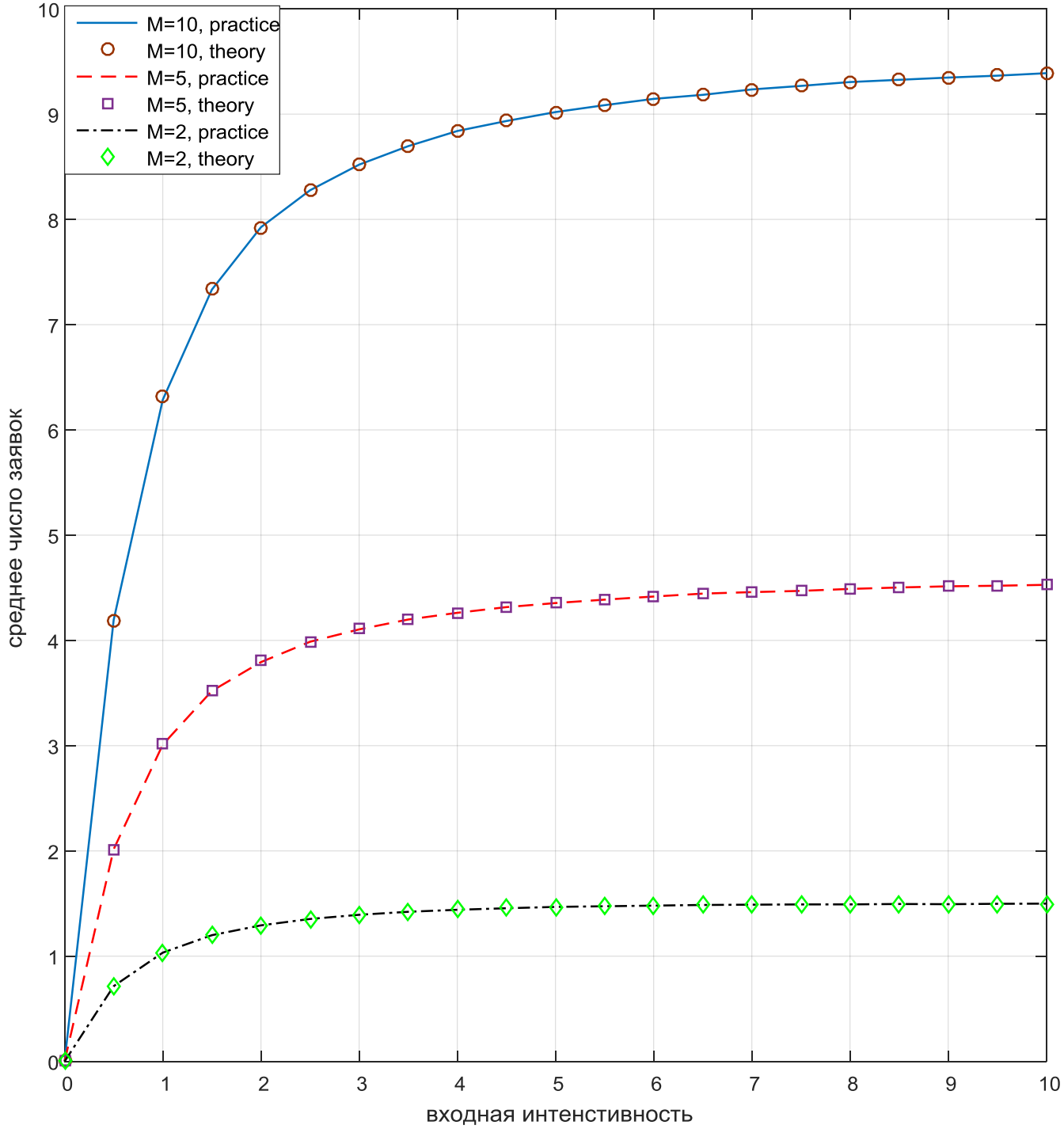

Рис. 5 - график зависимости среднего количества заявок в очереди от интенсивности входного потока практический и теоретический для 2, 5 и 10 абонентов

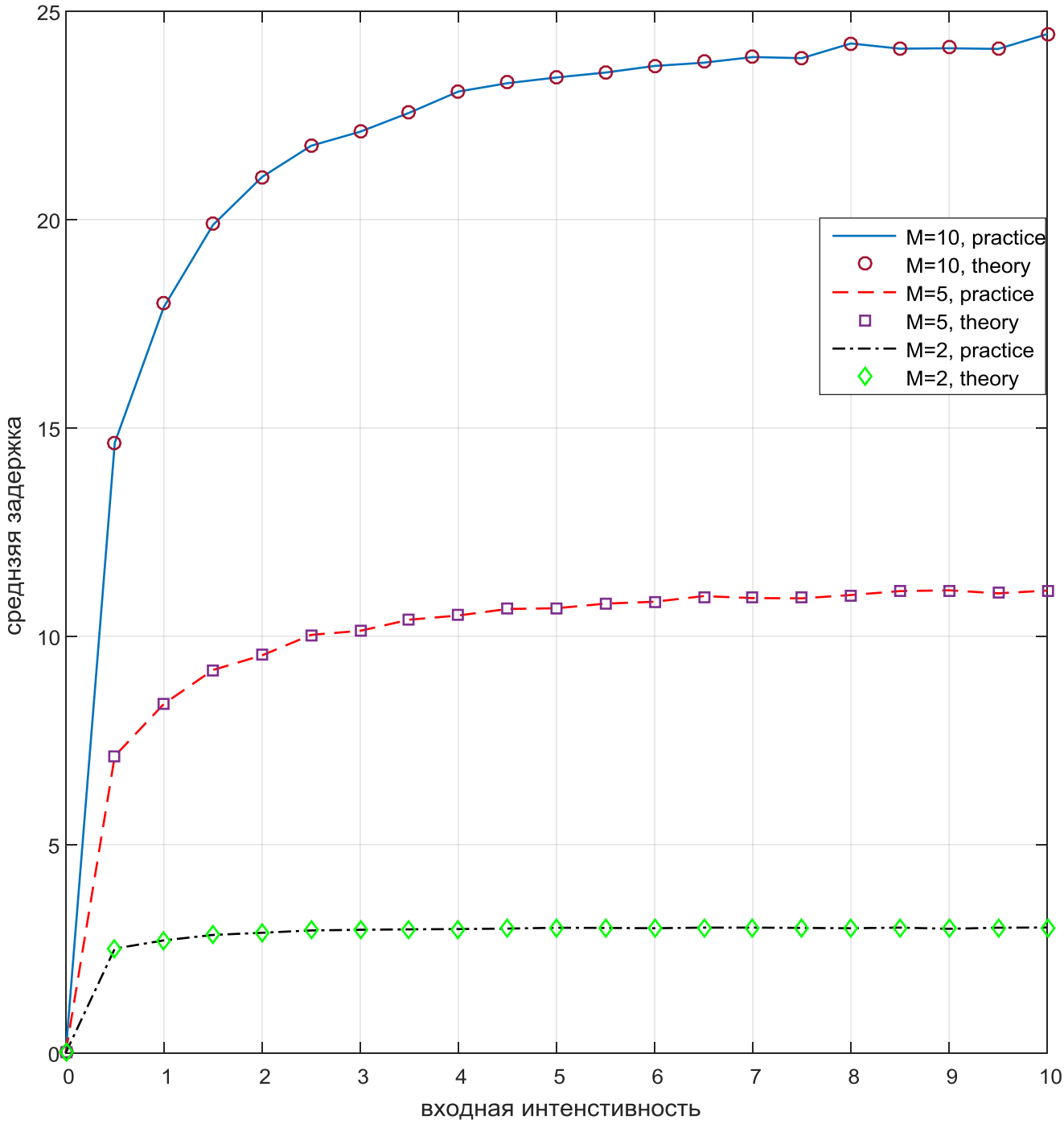

Рис. 6 - график зависимости задержки сообщения в системе от интенсивности входного потока практический и теоретический для 2, 5 и 10 абонентов

Вывод

В результате выполнения лабораторной работы было сделано следующее:

1. Проведен анализ характеристик алгоритма ALOHA с использованием аппарата Марковских цепей;
2. Выполнено имитационное моделирование алгоритма ALOHA;
3. Найдено среднее количество заявок в очереди и среднюю задержку сообщения в системе;
4. Найдена оценка среднего количества заявок в очереди и средней задержки с использованием Марковских цепей;
5. Построен график зависимости среднего количества заявок в очереди от интенсивности входного потока по результатам имитационного моделирования и оценки с использованием Марковских цепей;
6. Построен график зависимости средней задержки сообщения в системе от интенсивности входного потока по результатам имитационного моделирования и оценки с использованием Марковских цепей;
7. По полученным графикам можно сделать вывод о том, что имитационное моделирование и расчет совпадают, что говорит как о правильности работы моделирующей программы, так и о работоспособности метода расчета характеристик с использованием Марковских цепей;
8. По графику зависимости среднего количества заявок в очереди от интенсивности входного потока, можно сделать вывод о том, что при увеличении интенсивности входного потока, среднее количество заявок в очереди растет до определенного момента и после точки, так называемой критической, остается на постоянном уровне, и для данного алгоритма, этот уровень равен количеству абонентов в системе, так как, когда интенсивность потока будет очень большой, буфер у каждого абонента будет заполнен;
9. По графику зависимости средней задержки сообщения в системе от интенсивности входного потока, можно сделать вывод о том, что при увеличении интенсивности входного потока, средняя задержка растет до определенного уровня, и после критического значения интенсивности входного потока, остается на постоянном уровне, это можно объяснить тем, что с увеличением интенсивности входного потока, буфер абонентов будет заполнен, что будет приводить к

увеличенному количеству события КОНФЛИКТ, тем самым оставляя сообщение в системе более продолжительное время.

Список использованной литературы

1. Конспект лекций
2. Методические указания по лабораторной работе методы управления доступом к среде
3. http://www.scholarpedia.org/article/Aloha_random_access
4. http://www.wirelesscommunication.nl/reference/chaptr06/aloha/aloha.htm
5. http://sci.alnam.ru/book_otsp.php?id=105
6. Карлин С. Основы теории случайных процессов. М.: "Мир", 1971

ПРИЛОЖЕНИЕ

Листинг программы в среде matlab

```
close all;
clear all;
clc;
%% входные данные
T=100000;% количесвто окон
M=2;% количество абонентов
P_send=1/M;% вероятность передачи
lymda_init=0;
lymda_inc=0.5;% интенсивность входного потока
lymda_end=10;
B=1;% длина очереди
%% массивы для графиков
outLymda=zeros(1,length(lymda_init:lymda_inc:lymda_end));
meanQueu=zeros(1,length(lymda_init:lymda_inc:lymda_end));
meanZaderjka=zeros(1,length(lymda_init:lymda_inc:lymda_end));
N_teor1=zeros(1,length(lymda_init:lymda_inc:lymda_end));
N_teor=zeros(1,length(lymda_init:lymda_inc:lymda_end));
delay_teor=zeros(1,length(lymda_init:lymda_inc:lymda_end));
%% моделирование
count=1;
for lymda=lymda_init:lymda_inc:lymda_end
   disp(lymda);
%% построение матрицы переходов МЦ и получение теор. ср. числа
абонентов
[P,Matrics_2]  = create_P( M,lymda,P_send );% получили матрицу
переходов
C=P'-eye(M+1,M+1);
C(M+1,:)=1;
X=zeros(1,M+1);
X(M+1)=1;
Answ=X/C';% получили вектор стац. состояний
for i=0:1:M
    N_teor(1,count)=N_teor(1,count)+Answ(i+1)*i; %теоретическое
среднее число заявок
end
%% переменные для моделирования
Zayavki=poissrnd(lymda/M,M,T);% формирую поток заявок по окнам
queue=zeros(M,B); %очередь
sr_buffer=zeros(M,T);% среднее числов в буфере
Delay=zeros(M,1);% Задержка
Otpravka=zeros(M,1);% сколько ушло у абонента
When_sms_come=zeros(M,1);%когда пришло сообщение

    for t=1:T %% идем по окнам
%% смотрим заявки у абонентов и если есть кладем в очередь
```

```
        queue=queue+Zayavki(:,t);% смотрим есть ли заявки и
кладем в буфер
        queue(queue>B)=1;% подумать, если буфер больше 1
%% если она появилась записываем время
        if sum(queue)~=0 && sum(Zayavki(:,t))~=0
            for i=1:M
                if  When_sms_come(i)==0 && queue(i)~=0
                      When_sms_come(i)=t;  % записываем время
прибытия заявки
                end
            end
        end
%% есть те кто готов передать
        if sum(queue)~=0
          send_mes=zeros(M,1);
            for Abon=1:M
                p=rand;
                 if p<P_send % если случ величина больше вер
передачи, то можно передавать
                     if queue(Abon,1)~=0 &&  When_sms_come(
Abon,1)<t

                         send_mes(Abon,1)=1;%какого абонента
есть для передачи

                     end
                end
            end
        end
%% передача
        if sum(queue)~=0
            if sum(send_mes)==1
                [maxel,ind]=max(send_mes);
                if When_sms_come(ind,1)<t
                    Delay(ind,1)=Delay(ind,1)+(t-
When_sms_come(ind,1));%почситали задержку
                    Otpravka(ind,1)=Otpravka(ind,1)+1;%сколько
вышло
                    queue(ind,1)=queue(ind,1)-1;%очистили буфер(
если буфер не 1)
                    When_sms_come(ind,1)=0;%обнулили когда
пришла

                end
            end
        end
    sr_buffer(:,t)=queue;
    end


outLymda(1,count)=(sum(Otpravka))/T;      % выходная
интенсивность
```

```
meanQueu(1,count)=(sum(sum(sr_buffer)))/T; % среднее число
абонентов
meanZaderjka(1,count)=(sum(Delay))/(sum(Otpravka)); % средняя
задержка
delay_teor(1,count)=N_teor(1,count)/ outLymda(1,count); %
средняя задержка теория
count=count+1;
end
meanZaderjka(isnan(meanZaderjka))=0;
delay_teor(isnan(delay_teor))=0;
%% вывод графиков
 figure(1)
plot(lymda_init:lymda_inc:lymda_end,meanQueu);
  figure(2)
plot(lymda_init:lymda_inc:lymda_end,(meanZaderjka));
```

Функция для подсчета переходной матрицы Марковской цепи

```
function P = create_P( M,lymda,P_send )
y=exp((-1*lymda)/M);
P=zeros(M+1,M+1);
for i=0:M %по строке сверху вниз
    for j=0:M %по столбцам слева на право
        if i==0%переходы из 0 состояния
            if j==0
                P(i+1,j+1)=y^M;
            elseif j>0 && j<M
                P(i+1,j+1)= nchoosek(M,j)*(y^(M-j))*(1-y)^j;
            else
                 P(i+1,j+1)=(1-y)^M;
            end
        elseif i>0 && i<M
             if j==i-1
               P(i+1,j+1)=  i*P_send*((1-P_send)^(i-1))*(y^(M-
i));

             elseif j>i
                 a=j-i;
                  if a+1>M-i
                     P(i+1,j+1)=nchoosek(M-i,a)*((1-y)^a)*(y^(M-
i-a))*(1-i*P_send*((1-P_send)^(i-1)));
                  else
                 P(i+1,j+1)=nchoosek(M-i,a)*((1-y)^a)*(y^(M-i-
a))*(1-i*P_send*((1-P_send)^(i-1)))+...
                    + nchoosek(M-i,a+1)*((1-y)^(a+1))*(y^(M-i-
(a+1)))*i*P_send*((1-P_send)^(i-1));
                 end
                if j==M
                    P(i+1,i+1)=1-sum(P(i+1,:));
%                      v=y^(M-i)*nchoosek(M-i,1)*(1-y)*(y^(M-i-
1))*i*P_send*(1-P_send)^(i-1);
                end
```

```
            end
        else%случай M
            if j==i-1
                P(i+1,j+1)=M*P_send*((1-P_send)^(M-1));
            elseif j==M
                P(i+1,j+1)=1-M*P_send*((1-P_send)^(M-1));
            end
        end
    end
end
```