Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет

телекоммуникаций и информатики»

**ОТЧËТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

по дисциплине

«Теория функционирования распределённых вычислительных систем»

Выполнил:

Студент гр. ИС-641

Меняйлов В.О

Проверила:

Ассистент Кафедры ВС

Ткачева Т.А.

Новосибирск 2020

Оглавление

[**1.Цель роботы** 3](#_Toc37011755)

[**2.Теория** 5](#_Toc37011756)

[**2.1Производительность** 5](#_Toc37011757)

[**2.2Определения основных параметров и основные формулы** 5](#_Toc37011758)

[**3.Ход работы** 6](#_Toc37011759)

[**4.Заключение** 10](#_Toc37011760)

[**5. Листинг** 11](#_Toc37011761)

# **1.Цель роботы**

Имеется распределенная вычислительная система (ВС) укомплектованная *N* одинаковыми элементарными машинами (ЭМ). Основная подсистема (вычислительное ядро) ВС состоит из n ЭМ, *n – N* элементарных машин составляют структурную избыточность. Заданы *λ* – интенсивность потока отказов любой из *N* элементарных машин *([λ] = 1/ч)*, *m* – количество восстанавливающих устройств восстанавливающей системы и *µ* – интенсивность потока восстановления элементарных машин одним восстанавливающим устройством *([µ] = 1/ч)*.

В инженерной практике при анализе надежности ВС наиболее употребительный такие показатели как математическое ожидание времени безотказной работы (средней наработки до отказа) и среднего времени T восстановления ВС, которые равны:

Где R(t) – функция надежности ВС, а U(t) – функция восстановимости ВС.

Для распределенных ВС и T допустимо рассчитывать “частотным” методом [1] который обеспечивает результаты хорошо согласующиеся с более точными вычислениями:

1. Написать программу расчета частотным методом математического ожидания времени Ɵ безотказной работы и среднего времени T восстановления ВС со структурной избыточностью.
2. Построить графики зависимости значений показателя Θ от параметров λ, µ, m и n.

**2.1.** Построить график зависимости Ɵ(n). Параметры: N = 65536; λ = 10-5; m = 1; n = 65527, 65528, …, 65536; µ ∈ {1, 10, 100, 1000}

**2.2.** Построить график зависимости Ɵ(n). Параметры: N = 65536; µ = 1; m = 1; n = 65527, 65528, …, 65536; λ ∈ {10-5, 10-6, 10-7, 10-8, 10-9}.

**2.3.** Построить график зависимости Ɵ(n). Параметры: N = 65536; µ = 1;

λ = 10-5; n = 65527, 65528, …, 65536; m ∈ {1, 2, 3, 4}.

1. Построить графики зависимости значений показателя T от параметров λ, µ, m и n.

**3.1.** Построить график зависимости T(n). Параметры: N = 1000; λ = 10-3; m = 1; n = 900, 910, …, 1000; µ ∈ {1, 2, 4, 6}.

**3.2.** Построить график зависимости T(n). Параметры: N = 8192; µ = 1; m = 1; n = 8092, 8102, …, 8192; λ ∈ {10-5, 10-6, 10-7, 10-8, 10-9}.

**3.3.** Построить график зависимости T(n). Параметры: N = 8192; µ = 1; λ = 10-5; n = 8092, 8102, …, 8192; m ∈ {1, 2, 3, 4}.

# **2.Теория**

## **2.1Производительность**

Современные высокопроизводительные средства обработки информации – распределенные ВС ( distributed computer systems):

* Большемасштабность ( large-scale), массовый параллелизм(число процессоров
* Программируемость структуры (structure programmability).
* Масштабируемость (scalability)
* Мультипрограммный режим.

## **2.2Определения основных параметров и основные формулы**

– интенсивность потока отказов в любой из N машин.

- среднее время безотказной работы одной ЭМ

m , 1≤m≤N – размер восстанавливающей подсистемы

m- количество восстанавливающих устройств

µ - интенсивность потока восстановления (1/µ - обнаружение + восстановление).

Так же в инженерной практике наиболее употребительны не R(t) и U(t), а математическое ожидание времени безотказной работы ( средняя наработка до отказа и среднее время восстановления:

Ꜫ(t) – число исправных машин в момент времени t

- вероятность того, что в система начавшей функционировать в состоянии исправных машин

R(t) – функция надежности

U(t) – функция восстановимости

S(t) – Функция готовности

Среднее время безотказной работы ВС при n≠N и при n=N следующее:

Среднее время восстановления ВС при *n ≠ 1* и при *n = 1* , следующее

Из этих формул следует следующий вывод:

# **3.Ход работы**

Была написана программа для расчета частотным методом математического ожидания времени безотказно работы и среднего времени восстановления ВС со структурной избыточностью, каждый эксперимент запускался неоднократно, что бы получить более точные значения для наших экспериментов. По полученным значения были построены графики Зависимости показателей.

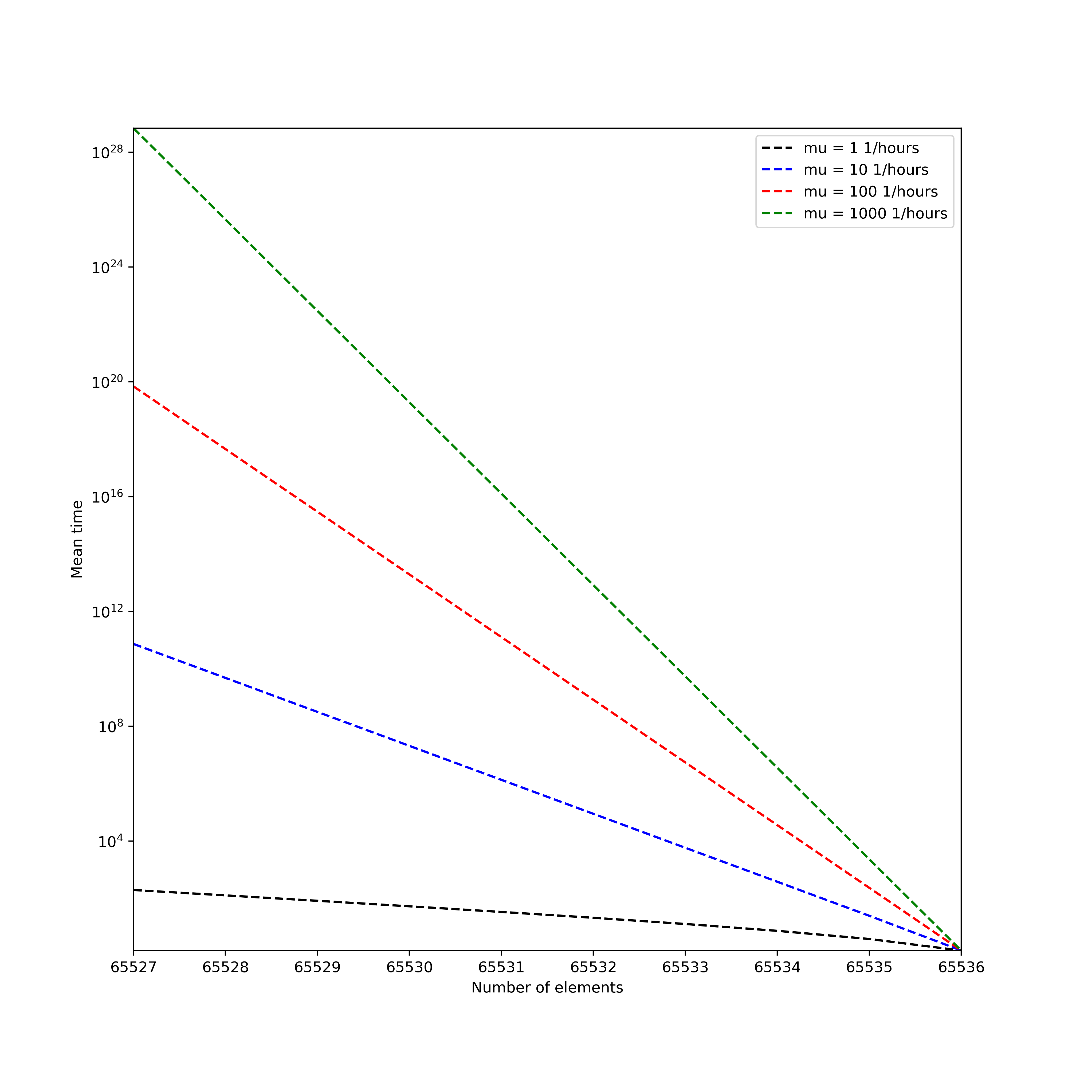


Рисунок 1- Графим зависимости от значения mu

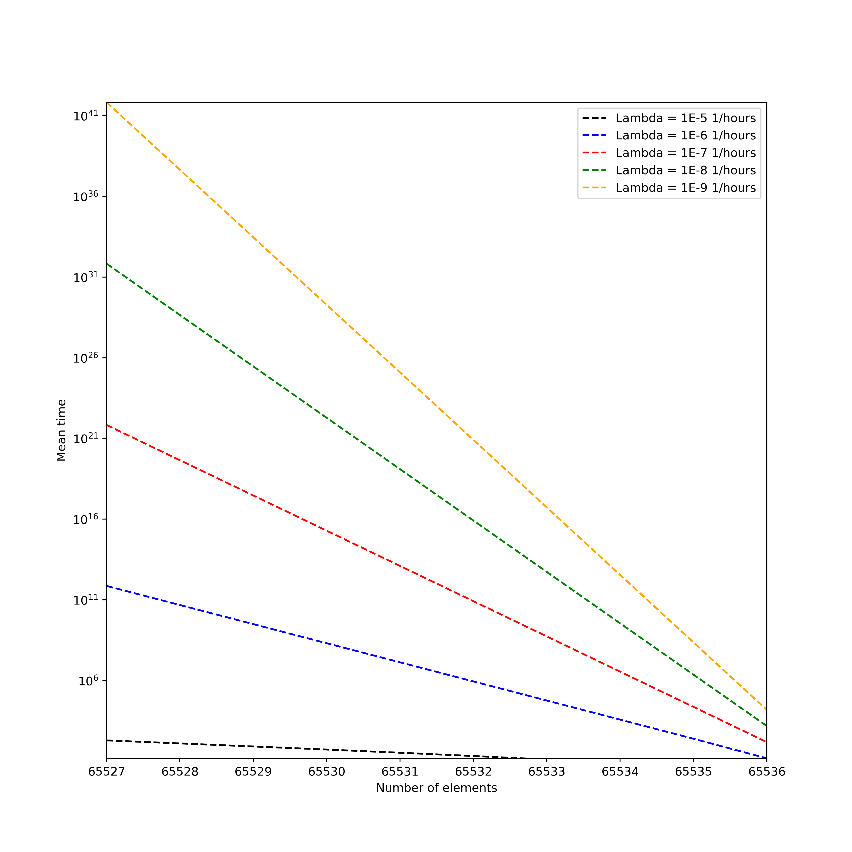


Рисунок 2- График зависимости от значения Lambda

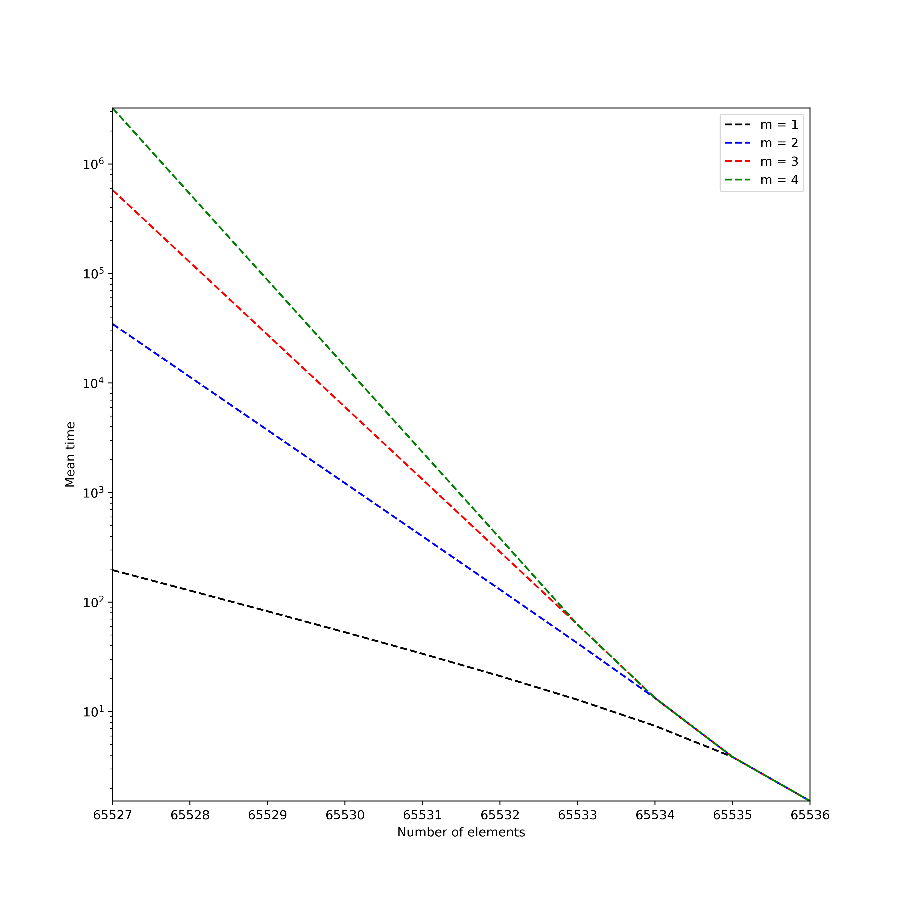


Рисунок - График зависимости от значения m

Для более удобного наблюдения поведения нашего графика было принято прологарифмировать ось Y. Можно заметить нетипичное поведение при изменении интенсивности потока отказов, кол-ва восстанавливающих устройств и интенсивности потока восстановления.

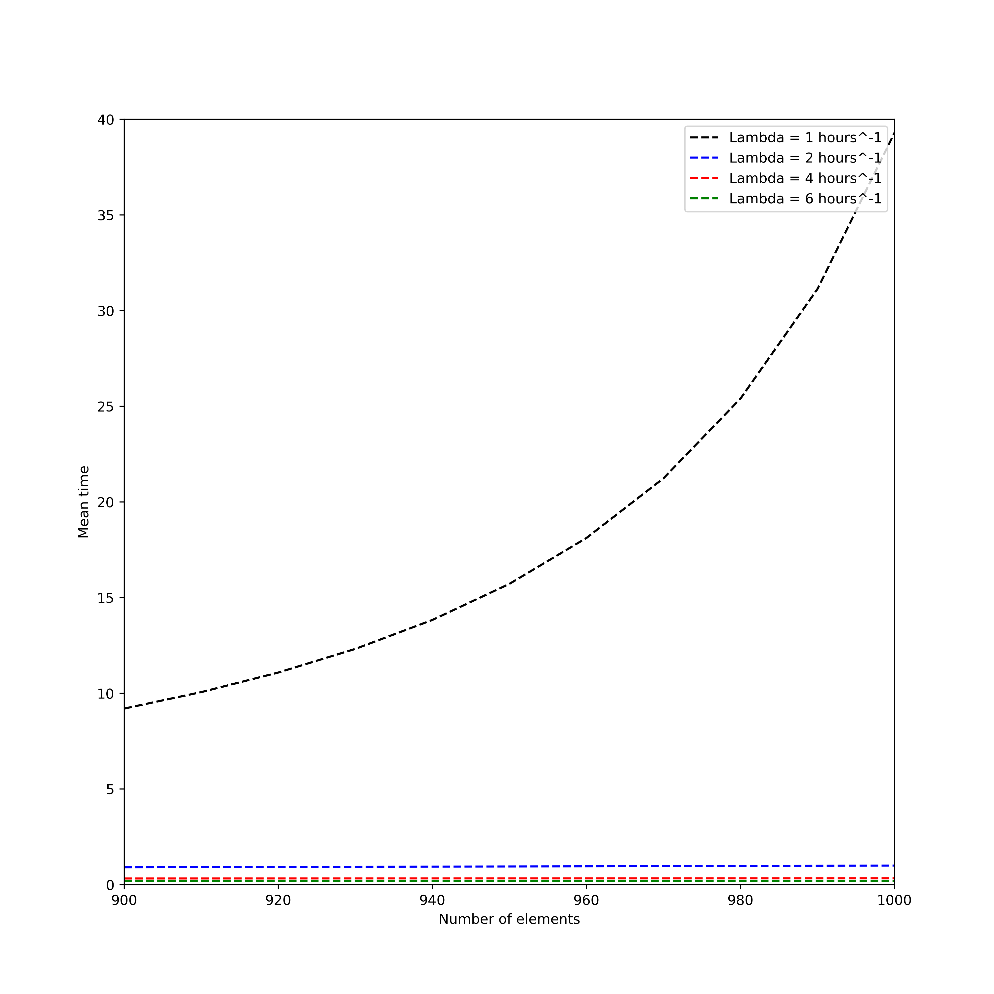


Рисунок 4- График зависимости mu

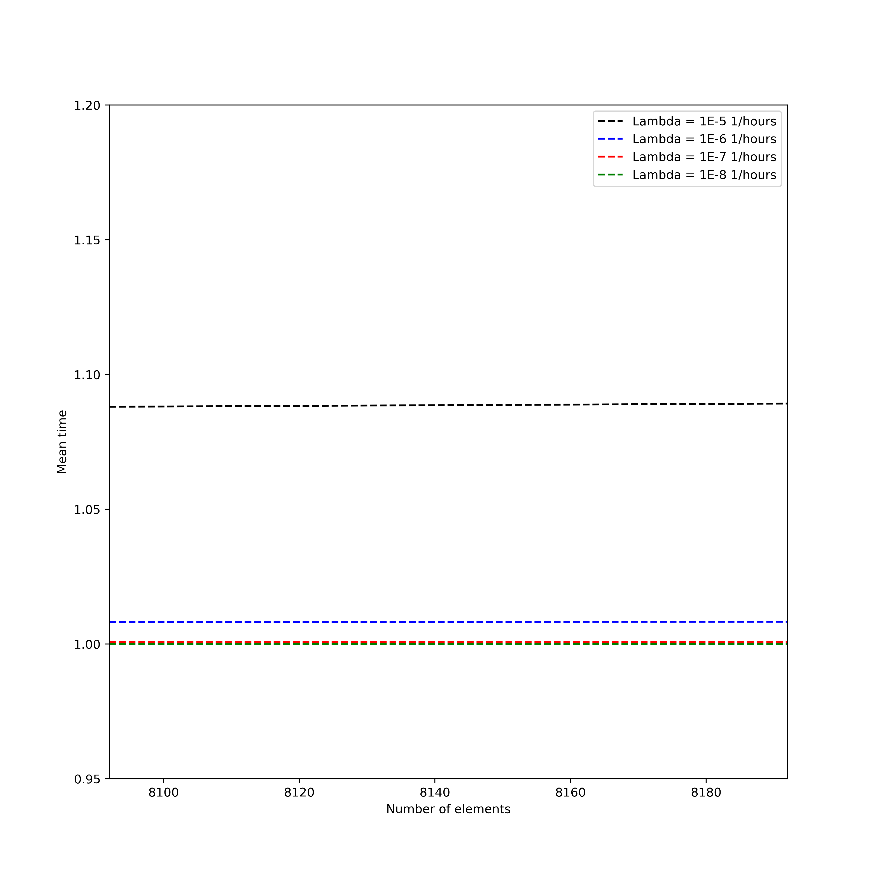


Рисунок 5- График зависимости от значения lambda

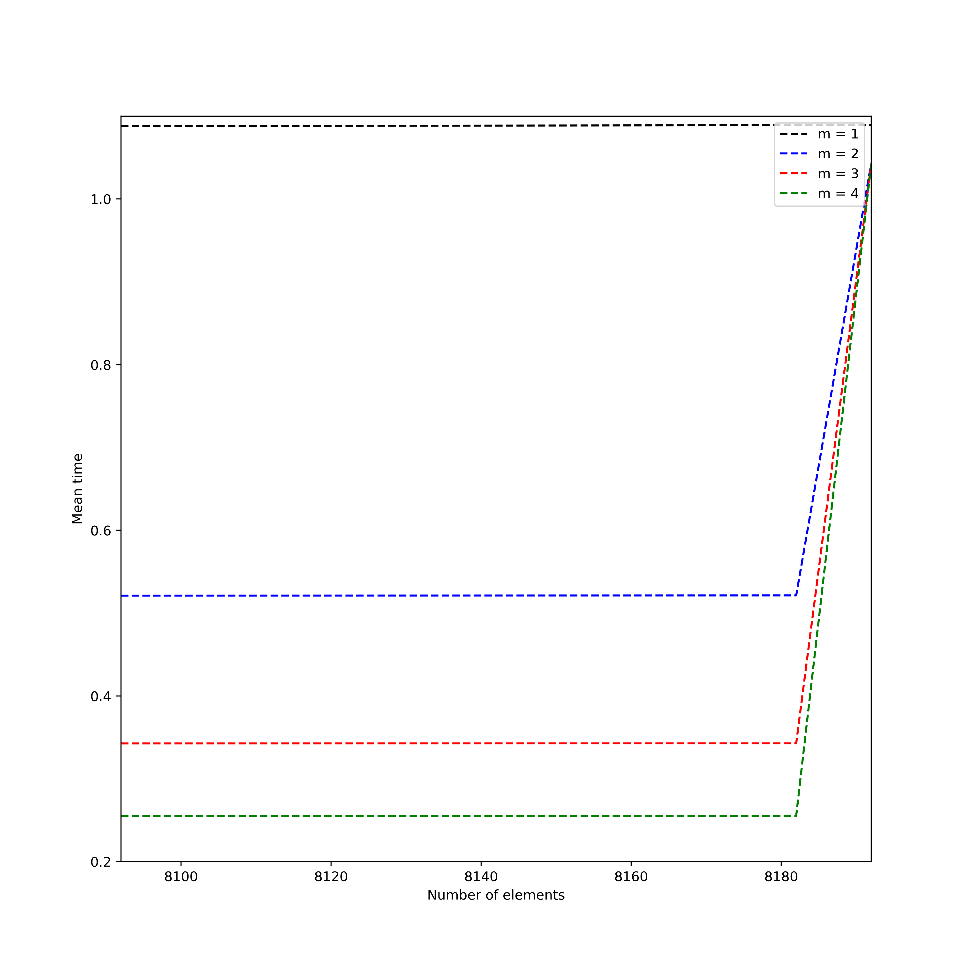


Рисунок 6- График зависимости от значения m

Зависимость показателя T от параметров интенсивности потока отказов, кол-ва восстанавливающих устройств и интенсивности потока восстановления, так же показали не типичное поведение, то есть в некоторых случаях график себя ведет по особенному.

# **4.Заключение**

В результате лабораторной работы была написана программа, которая рассчитала частотным методом математического ожидание безотказной работы и среднего времени T восстановления ВС со структурной избыточностью.

По графикам можно сделать вывод о том, что время между неисправности будет возрастать при увеличении интенсивности потока отказов, но увеличивая количество восстанавливающих устройств , мы имеем возможность увеличить время между неисправностями ВС. Это все справедливо так же и для интенсивности потока восстановления, при его увеличении. При увеличении среднего времени восстановления, мы уменьшим время восстановления ВС.

# **5. Листинг**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

double Theta(double lambda, double mu, int n, int N, int m)

{

double totalSum = 0.0f;

double totalMul = 1.0f;

for (int j = n + 1; j <= N; ++j) {

double mul = (j - 1 >= N - m && j - 1 <= N) ? (N - (j - 1)) \* mu : m \* mu;

totalMul \*= mul / ((j - 1) \* lambda);

totalSum += totalMul / (j \* lambda);

}

return totalSum + 1 / (n \* lambda);

}

double Tau(double lambda, double mu, int n, int N, int m)

{

if (n == 1)

return m \* mu;

double totalMul = 1.0f;

for (int l = 1; l <= n - 1; ++l)

totalMul \*= l \* lambda / (mu \* l);

double totalSum = 0.0f;

for (int j = 1; j <= n - 1; ++j) {

double totalMul = 1.0f;

for (int l = j; l <= n - 1; ++l) {

double mul = (l >= N - m && l <= N) ? (N - l) \* mu : m \* mu;

totalMul \*= l \* lambda / mul;

}

totalSum += totalMul / (j \* lambda);

}

return totalMul + totalSum;

}

int main(int argc, char\* argv[])

{

double lambda = 0.00001;

int N = 65536;

int m = 1;

FILE\* f\_out = fopen("1.dat", "w");

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int mu = 1; mu != 10000; mu \*= 10)

fprintf(f\_out, "\"mu = %d 1/hours\" ", mu);

fprintf(f\_out, "\n");

for (int n = 65527; n <= N; ++n) {

fprintf(f\_out, "%d ", n);

for (int mu = 1; mu != 10000; mu \*= 10)

fprintf(f\_out, "%.6f ", Theta(lambda, mu, n, N, m));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

f\_out = fopen("2.dat", "w");

double mu = 1.0;

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int i = 5; i != 10; ++i)

fprintf(f\_out, "\"Lambda = 1E-%d 1/hours\" ", i);

fprintf(f\_out, "\n");

for (int n = 65527; n <= N; ++n) {

fprintf(f\_out, "%d ", n);

for (int i = -5; i >= -9; --i)

fprintf(f\_out, "%.6f ", Theta(pow(10.0, i), mu, n, N, m));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

f\_out = fopen("3.dat", "w");

lambda = 0.00001;

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int i = 1; i != 5; ++i)

fprintf(f\_out, "\"m = %d\" ", i);

fprintf(f\_out, "\n");

for (int n = 65527; n <= N; ++n) {

fprintf(f\_out, "%d ", n);

for (int i = 1; i <= 4; ++i)

fprintf(f\_out, "%.6f ", Theta(lambda, mu, n, N, i));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

f\_out = fopen("4.dat", "w");

lambda = 0.001;

N = 1000;

int mus[] = { 1, 2, 4, 6 };

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int i = 0; i != 5; ++i)

fprintf(f\_out, "\"Lambda = %d hours^-1\" ", mus[i]);

fprintf(f\_out, "\n");

for (int n = 900; n <= N; n += 10) {

fprintf(f\_out, "%d ", n);

for (int i = 0; i < 4; ++i)

fprintf(f\_out, "%.6f ", Tau(lambda, mus[i], n, N, m));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

f\_out = fopen("5.dat", "w");

mu = 1.0;

N = 8192;

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int i = 5; i != 10; ++i)

fprintf(f\_out, "\"Lambda = 1E-%d 1/hours\" ", i);

fprintf(f\_out, "\n");

for (int n = 8092; n <= N; n += 10) {

fprintf(f\_out, "%d ", n);

for (int i = -5; i>= -9; --i)

fprintf(f\_out, "%.6f ", Tau(pow(10.0, i), mu, n, N, m));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

f\_out = fopen("6.dat", "w");

lambda = 0.00001;

N = 8202;

N = 8192;

if (f\_out) {

fprintf(f\_out, "\"Number n of elementary machines in base subsystem\" ");

for (int i = 1; i != 5; ++i)

fprintf(f\_out, "\"m = %d\" ", i);

fprintf(f\_out, "\n");

for (int n = 8092; n <= N; n += 10) {

fprintf(f\_out, "%d ", n);

for (int i = 1; i <= 4; ++i)

fprintf(f\_out, "%.6f ", Tau(lambda, mu, n, N, i));

fprintf(f\_out, "\n");

}

fclose(f\_out);

}

return EXIT\_SUCCESS;

}