

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Моделирование непрерывных и дискретных систем»

на тему «Имитационное моделирование распределенного
приложения для компьютерной сети»

Выполнил:

студент группы 21ВВВ1

Гришин С.В.

Принял:

д.т.н., профессор Зинкин С.А.

к.т.н., доцент Федюнин Р.Н.

Пенза 2024

Содержание

Введение.....	3
1. Теоретическая часть задания	4
1.1 Описание компьютерных сетей.....	4
1.2 Сетевая модель TCP/IP и сеть Ethernet	5
2. Моделирование сети	6
2.1 Структура моделируемой сети	6
2.2 Результаты моделирования на Python.....	7
2.3 Разработка алгоритма моделирования.....	9
2.4 Результаты моделирования в среде Pipe	12
Заключение	14
Список используемых источников.....	15
Приложение А	16
Приложение В	18
Приложение С	23

Введение

Главной целью данного курсового проекта является разработка имитационной модели передачи сообщений в сети с учетом таких параметров, как задержки, формирование очередей и скорость передачи пакетов по каналам связи. Для моделирования системы были выбраны программы GPSS World и PIPE.

GPSS World представляет собой универсальную среду, предназначенную для моделирования как дискретных, так и непрерывных процессов, что делает ее подходящей для профессионального моделирования различных систем и процессов. Эта система стала следующей версией GPSS/PC, выпущенной в 1984 году для DOS. Обе версии были разработаны специалистами компании Minuteman Software, основанной в 1982 году под руководством Спрингера Кокса. Изначально GPSS World была выпущена в 1994 году для платформы OS/2 от IBM, а в 2000 году была адаптирована для операционной системы Windows от Microsoft.

1. Теоретическая часть задания

1.1 Описание компьютерных сетей

Компьютерная сеть — это система компьютеров, соединенных проводными или беспроводными технологиями для обмена данными и ресурсами, а также для обеспечения общего доступа. Ниже приведены наиболее распространенные виды компьютерных сетей:

LAN (локальная сеть): локальная сеть объединяет компьютеры на небольших расстояниях, что позволяет обмениваться файлами, данными и ресурсами. Например, LAN может охватывать компьютеры в офисе, учебном заведении или больнице. Эти сети обычно принадлежат и управляются частными владельцами.

WLAN (беспроводная локальная сеть): WLAN аналогична локальной сети, но устройства подключаются беспроводным способом.

WAN (глобальная сеть): WAN соединяет компьютеры на большом расстоянии, иногда охватывая несколько регионов или континентов. Интернет — крупнейшая WAN-сеть, объединяющая миллиарды устройств по всему миру. Управление WAN часто основано на совместном или распределенном владении.

MAN (городская сеть): сети MAN обычно принадлежат муниципалитетам и государственным организациям.

CAN (кампусная сеть): CAN, также известная как корпоративная сеть, обслуживает университеты, бизнес-центры и другие крупные учреждения.

VPN (виртуальная частная сеть): VPN — это безопасное соединение между двумя точками, создающее зашифрованный канал для защиты данных и идентификационной информации пользователя от постороннего доступа.

1.2 Сетевая модель TCP/IP и сеть Ethernet

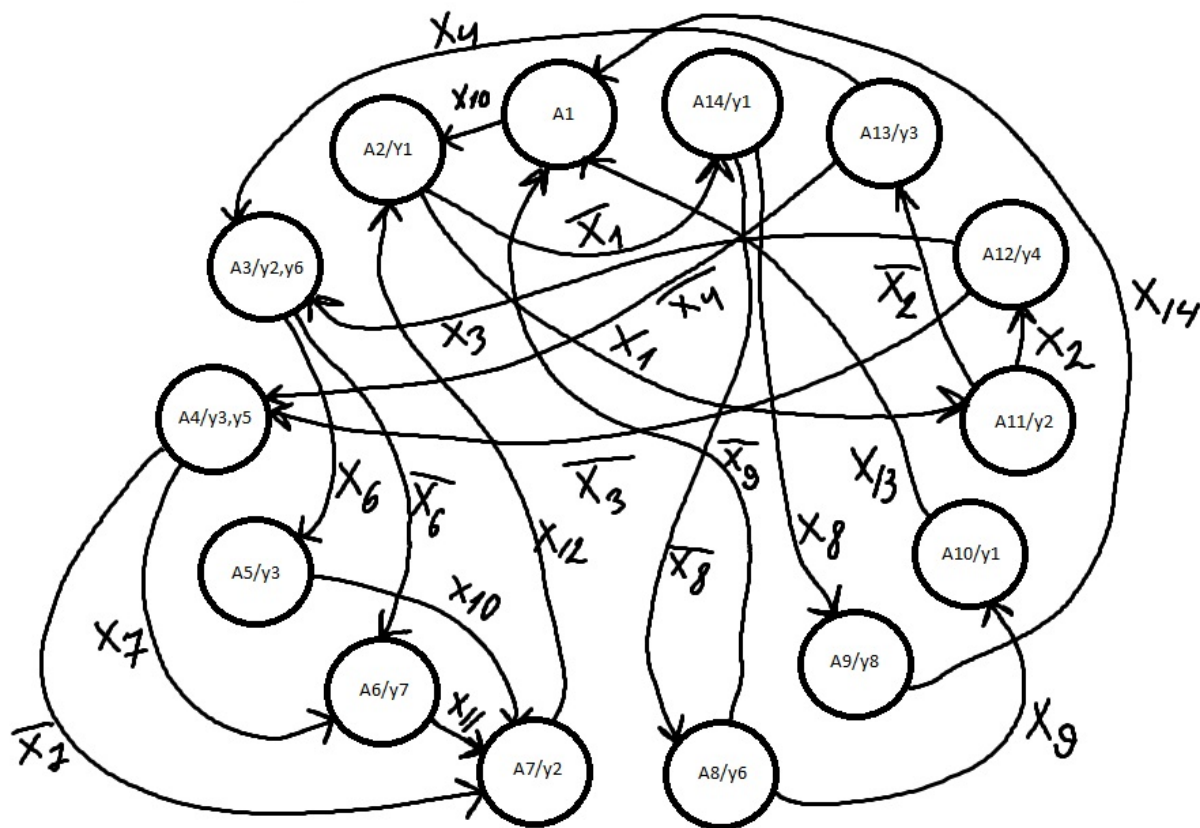
Стек протоколов TCP/IP (Transmission Control Protocol/Internet Protocol) представляет собой сетевую модель, описывающую передачу цифровых данных. Модель получила название от двух основных протоколов, на которых построен интернет. В 1970-х годах передача данных между разными сетями была невозможна, и для решения этой задачи был создан стек интернет-протоколов TCP/IP.

Протоколы разрабатывались Министерством обороны США, из-за чего модель TCP/IP иногда называют моделью DoD (Department of Defense). Модель TCP/IP структурирована в виде нескольких уровней, каждый из которых включает определенные протоколы и выполняет свои функции.

Набор протоколов TCP/IP обеспечивает сквозную передачу данных, определяя, как данные должны быть упакованы, обработаны, переданы, маршрутизированы и получены. Эта функциональность разделена на четыре уровня абстракции, распределяющих протоколы в зависимости от масштабов сетевого взаимодействия. От самого низкого к самому высокому уровням они включают: уровень связи, отвечающий за взаимодействие данных в пределах одного сегмента сети; интернет-уровень, который поддерживает связь между независимыми сетями; транспортный уровень, обеспечивающий связь между хостами; и прикладной уровень, который организует обмен данными для приложений.

2. Моделирование сети

2.1 Структура моделируемой сети



Граф коммуникаций с проставленными именами узлов и каналов передачи данных показан на рисунке 1.

Рисунок 1 - Граф с именами узлов и каналами передачи

Граф коммуникаций с отмеченными узлами, на которых будут выполняться операторы (программные модули) алгоритма показан на рисунке 2.

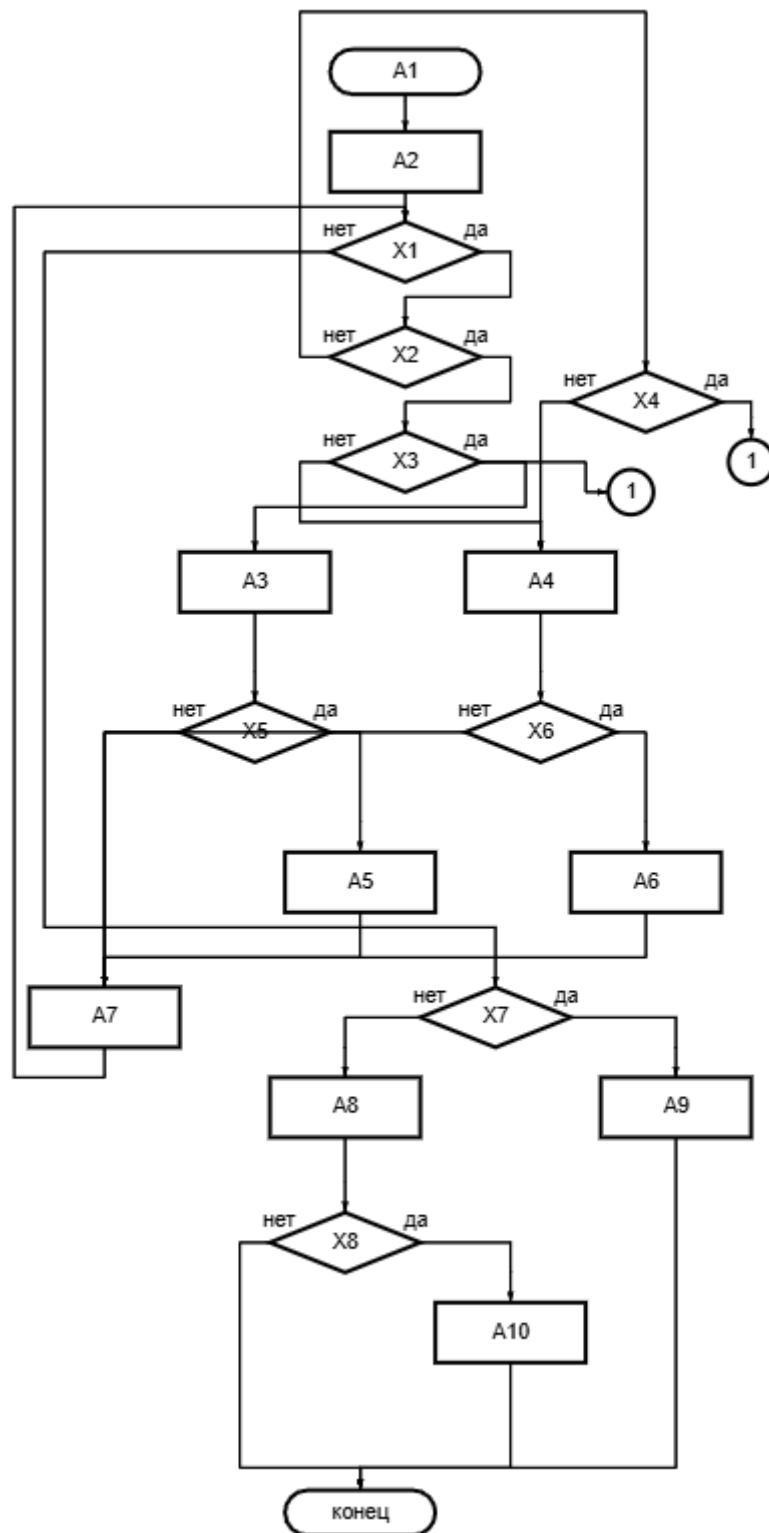


Рисунок 2 - Граф с отмеченными узлами

Структура компьютерной сети - ТСР/IP сеть с отмеченными сетями и маршрутизаторами с интерфейсами (у каждого интерфейса имеется свой IP

адрес). Через данные интерфейсы к маршрутизаторами подключаются сети. (Приложение С.1).

2.2 Результаты моделирования на C++

Для моделирования работы автомата Мили была создана программа на языке программирования C++, представленная ниже.

```
using System;
using System.Linq;
using System.Threading;

namespace lb2_zCs
{
    internal class Program
    {
        static bool MoveVisual;
        static void Main(string[] args)
        {
            Console.CursorVisible = false;

            MoveVisual = true;
            int CurrentState = 1;
            string Steps = "A" + CurrentState;

            string[] A = { "1-2/10-0",
                           "2-14/!1,11/1-y1",
                           "3-5/6,6/!6-y6",
                           "4-6/7,7/!7-y3,y5",
                           "5-7/10-y3",
                           "6-7/11-y7",
                           "7-2/12-y2",
                           "8-1/!9,10/9-y6",
                           "9-1/14-y8",
                           "10-1/13-y1",
                           "11-12/2,13/!2-y2",
```

```

"12-4/!3,3/3-y4",
"13-4/!4,3/4-y3",
"14-9/8,8/!8-y1"};

```

```

while(true)
{
    Console.ForegroundColor = ConsoleColor.Cyan;
    Console.WriteLine(Steps);
    Thread.Sleep(1000);
    CurrentState = CulcState(A[CurrentState - 1]);
    if (!MoveVisual) Steps = Steps + "!";
    Steps = Steps + $">A{CurrentState}";
    Console.SetCursorPosition(0, 0);
}
}
public static int CulcState(string State)
{
    string[] StateParts = State.Split('-');
    string[] StateResult = StateParts[1].Split(',');
    if (StateResult.Length == 1)
    {
        return
int.Parse(StateResult[0].Split('/')[0].TrimStart('!'));
    }
    else
    {
        if (!Ask($"X" + StateResult[0].Split('/')[1].Trim('!')))
        {
            if (StateResult[0].Split('/')[1].Contains('!'))
            {
                return
int.Parse(StateResult[0].Split('/')[0].TrimStart('!'));
            }
            else
            {

```

```

        return
int.Parse(StateResult[1].Split('/')[0].TrimStart('!'));
    }
}
else
{
    if (!StateResult[0].Split('/')[1].Contains('!'))
    {
        return
int.Parse(StateResult[0].Split('/')[0].TrimStart('!'));
    }
    else
    {
        return
int.Parse(StateResult[1].Split('/')[0].TrimStart('!'));
    }
}
}
}
public static bool Ask(string str)
{
    while (true)
    {
        Console.ForegroundColor = ConsoleColor.DarkYellow;
        Console.Write("\nВыберите " + str + " \nY - да, N -
нет");

        ConsoleKey consoleKey = Console.ReadKey().Key;
        Console.SetCursorPosition(0, Console.CursorTop-1);
        Console.Write(new String(' ', Console.BufferWidth*2));
        if (consoleKey == ConsoleKey.Y)
        {
            MoveVisual = true;
            return true;
        }
        if (consoleKey == ConsoleKey.N)
        {
            MoveVisual = false;

```

```

        return false;
    }
}

}

}
}

```

Этот код реализует конечный автомат, где каждое состояние имеет определенный выходной сигнал и переход в другое состояние в зависимости от входного сигнала ('0' или '1'). Начальное состояние — 'a1', которое имеет выход 'Y1'. При каждом шаге автомат выводит текущее состояние и соответствующий выход, затем запрашивает у пользователя ввод ('0' или '1') для переменной, связанной с текущим состоянием. На основе введенного значения происходит переход в новое состояние, определяемое таблицей переходов transitions.

Результат работы программы представлен на рисунке 3.

Рисунок 3 – Результат работы программы

2.3 Разработка алгоритма моделирования

Выполнение оператор на узле реализуется следующим образом:

```

MA_1  SEIZE      MAA_1
      ADVANCE    1

```

```
RELEASE      MAA_1
```

Формирование и проверка логического условия формируется следующей последовательностью

```
MB_2  SEIZE      MBB_2
      ADVANCE    1
      ASSIGN     1, FN$COND
      RELEASE    MBB_2
      TEST E     P1, 1, MB_14
MB_11  SEIZE      MBB_11
      ADVANCE    1
      ASSIGN     1, FN$COND
      RELEASE    MBB_11
      TEST E     P1, 1, MB_13
```

Для формирования значение параметра P1 текущего транзакта, используются следующие блоки

```
COND  FUNCTION RN1, D2
      0.5, 0/1.0, 1
GENERATE 100
```

Полная программа моделирования представлена в приложении А.

2.3 Результаты моделирования в среде GPSS

Результаты моделирования показаны на рисунках 4-6 в приложение В. По результатам моделирования сделан вывод, что программа составлена верно. На блоке Generate было сгенерировано 100 транзактов, которые с учетом логических условий прошли по всем узлам через каналы передачи данных. На операторе TERMINATE кол-во транзактов также равняется 1000, что означает успешное удаление всех транзактов.

Задана архитектура сети, содержащая узлы и каналы передачи данных. В данной сети передаются пакеты. Размер пакета 64 КБайта.

При передаче данных исходные сообщения разбиваются на пакеты. Скорость передачи данных по каналу составляет 2 Мбит / С.

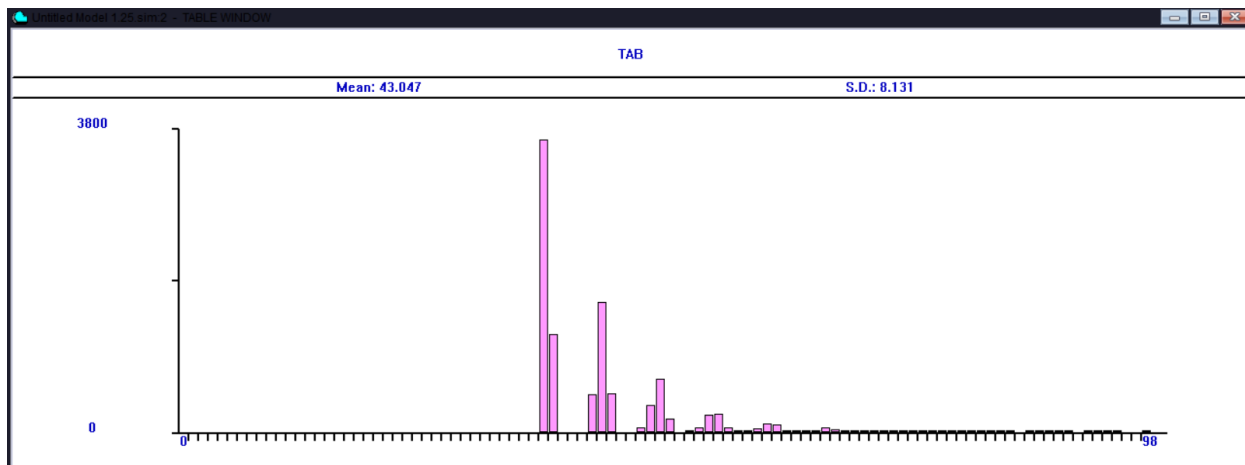


Рисунок 4 - При ADVANCE 32

2.4 Результаты моделирования в среде PIPE

Согласно заданию, была построена сеть Петри в программе Pipe (рисунок 7).

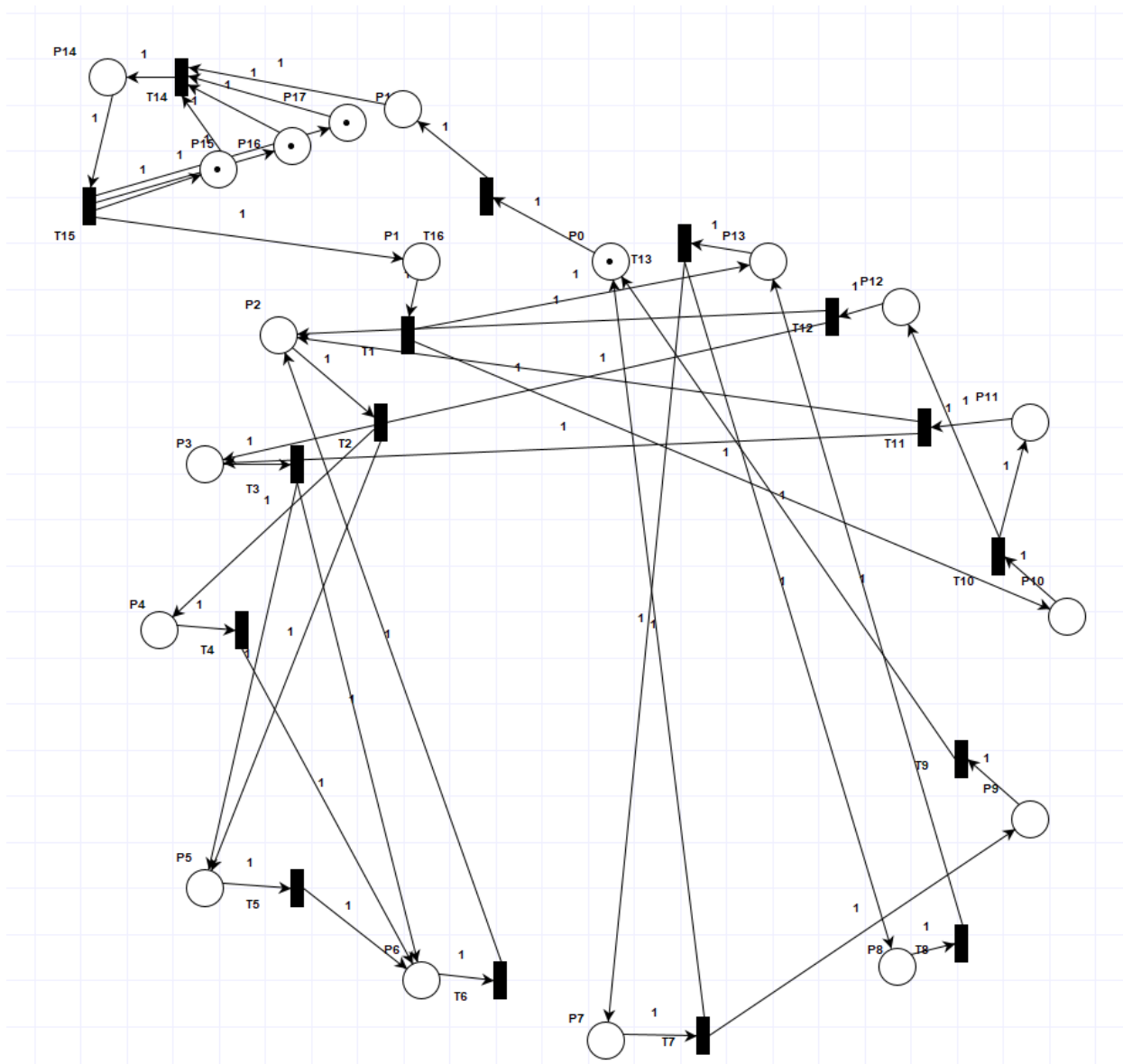


Рисунок 5 - Сеть Петри

По результатам проверки сети Петри, сделан вывод о правильности её конфигурации (рисунок 8).

Petri net state space analysis results

Bounded	true
Safe	true
Deadlock	false

Рисунок 6 - Результаты анализа сети

Animation history

Initial Marking

T16

T14

T15

T4

T19

T22

T7

T11

T4

T20

T23

T8

T10

T11

T3

T25

T17

T16

T14

T15

T4

T20

T24

T6

T9

T11

Рисунок 8 – Результаты моделирования сети

На основании этих результатов можно сделать вывод, что построенная сеть Петри работает корректно.

Заключение

В процессе выполнения курсовой работы были приобретены практические навыки работы с языком моделирования GPSS-WORLD, а также изучены его ключевые операторы и особенности.

На основе заданного графа коммуникаций была разработана и реализована структура сети, в которой каждому узлу и каждому каналу передачи данных были присвоены уникальные имена. В рамках работы была создана имитационная модель, позволяющая наглядно отследить процесс передачи сообщений по сети. Модель позволила детально проследить траекторию транзактов и особенности их прохождения через различные узлы

и блоки сети. На основе проведенного моделирования был сформирован подробный отчет, содержащий полную информацию о движении транзактов в системе, прохождении ими различных этапов обработки и влиянии на общую пропускную способность сети.

Список используемых источников

1. Кудрявцев Е. М. GPSS World. Основы имитационного моделирования различных систем. / Е. М.. Кудрявцев. - М.: ДМК Пресс, 2004. – 320 с.
2. Е.В. Смирнова, П.В. Козик. Технологии современных сетей Ethernet. Методы коммутации и управления потоками данных. - СПб.: БХВПетербург, 2012. - 272 с.
3. Томашевский В., Жданова Е. Имитационное моделирование в среде GPSS. / В. Томашевский, Е. Жданова – М.: Бестселлер, 2003. – 416 с.
4. Шеннон Р. Имитационное моделирование систем – искусство и наука.
5. В.Д. Боев. Моделирование систем. Инструментальные средства GPSS World. - СПб.: БХВ-Петербург, 2004. - 368 с.

Приложение А

Листинг программы

```

COND      FUNCTION RN1,D2
          0.5,0/1.0,1
          GENERATE 100

MA_1      SEIZE      MAA_1
          ADVANCE    1
          RELEASE    MAA_1
MET1      GATE NU YZ1
          GATE NU KAN1
          GATE NU YZ2
          TRANSFER SIM,,MET1
          SEIZE YZ1
          SEIZE KAN
          SEIZE YZ2
          ADVANCE 32
          RELEASE YZ1
          RELEASE KAN
          RELEASE YZ2
          TRANSFER ,MB_2

MB_2      SEIZE      MBB_2
          ADVANCE    1
          ASSIGN     1,FN$COND
          RELEASE    MBB_2
          TEST E     P1,1,MB_14
MB_11     SEIZE      MBB_11
          ADVANCE    1
          ASSIGN     1,FN$COND
          RELEASE    MBB_11
          TEST E     P1,1,MB_13
MB_12     SEIZE      MBB_12
          ADVANCE    1
          ASSIGN     1,FN$COND
          RELEASE    MBB_12
          TEST E     P1,1,MB_4
MB_3      SEIZE      MBB_3
          ADVANCE    1
          ASSIGN     1,FN$COND
          RELEASE    MBB_3
          TEST E     P1,1,MA_6
MA_5      SEIZE      MAA_5
          ADVANCE    1
          RELEASE    MAA_5
MA_7      SEIZE      MAA_7
          ADVANCE    1
          RELEASE    MAA_7
          TRANSFER   ,MB_2
MB_14     SEIZE      MBB_14
          ADVANCE    1
          ASSIGN     1,FN$COND
          RELEASE    MBB_14
          TEST E     P1,1,MB_8
MA_9      SEIZE      MAA_9
          ADVANCE    1
          RELEASE    MAA_9
          TRANSFER   ,ME_1
MB_13     SEIZE      MBB_13
          ADVANCE    1
          ASSIGN     1,FN$COND

```

	RELEASE	MBB_13
	TEST E	P1,1,MB_4
	TRANSFER	,MB_3
MA_10	SEIZE	MAA_10
	ADVANCE	1
	RELEASE	MAA_10
	TRANSFER	,ME_1
MB_4	SEIZE	MBB_4
	ADVANCE	1
	ASSIGN	1, FN\$COND
	RELEASE	MBB_4
	TEST E	P1,1,MA_7
MA_6	SEIZE	MAA_6
	ADVANCE	1
	RELEASE	MAA_6
	TRANSFER	,MA_7
MB_8	SEIZE	MBB_8
	ADVANCE	1
	ASSIGN	1, FN\$COND
	RELEASE	MBB_8
	TEST E	P1,1,ME_1
	TRANSFER	,MA_10
ME_1	SEIZE	ENDs
	ADVANCE	1
	RELEASE	ENDs
met	TABULATE	tab
	TERMINATE	1
tab	TABLE	M1,0,1,100
	START	10000

Приложение В

Результаты моделирования

GPSS World Simulation Report - Untitled Model 1.26.1

Thursday, December 05, 2024 12:37:49

START TIME	END TIME	BLOCKS	FACILITIES	STORAGES
0.000	1000055.000	82	19	0

NAME	VALUE
COND	10000.000
ENDS	10018.000
KAN	10006.000
KAN1	10004.000
MAA_1	10002.000
MAA_10	10017.000
MAA_5	10019.000
MAA_6	10011.000
MAA_7	10012.000
MAA_9	10020.000
MA_1	2.000
MA_10	59.000
MA_5	37.000
MA_6	68.000
MA_7	40.000
MA_9	49.000
MBB_11	10008.000
MBB_12	10009.000
MBB_13	10013.000
MBB_14	10015.000
MBB_2	10007.000
MBB_3	10010.000
MBB_4	10014.000
MBB_8	10016.000

MB_11	22.000
MB_12	27.000
MB_13	53.000
MB_14	44.000
MB_2	17.000
MB_3	32.000
MB_4	63.000
MB_8	72.000
MET	81.000
MET1	5.000
ME_1	78.000
TAB	10001.000
YZ1	10003.000
YZ2	10005.000

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT	COUNT	RETRY
	1	GENERATE	10000		0	0
MA_1	2	SEIZE	10000		0	0
	3	ADVANCE	10000		0	0
	4	RELEASE	10000		0	0
MET1	5	GATE	10000		0	0
	6	GATE	10000		0	0
	7	GATE	10000		0	0
	8	TRANSFER	10000		0	0
	9	SEIZE	10000		0	0
	10	SEIZE	10000		0	0
	11	SEIZE	10000		0	0
	12	ADVANCE	10000		0	0
	13	RELEASE	10000		0	0
	14	RELEASE	10000		0	0
	15	RELEASE	10000		0	0
	16	TRANSFER	10000		0	0
MB_2	17	SEIZE	20085		0	0
	18	ADVANCE	20085		0	0
	19	ASSIGN	20085		0	0
	20	RELEASE	20085		0	0
	21	TEST	20085		0	0
MB_11	22	SEIZE	10085		0	0
	23	ADVANCE	10085		0	0
	24	ASSIGN	10085		0	0
	25	RELEASE	10085		0	0
	26	TEST	10085		0	0
MB_12	27	SEIZE	5032		0	0
	28	ADVANCE	5032		0	0
	29	ASSIGN	5032		0	0
	30	RELEASE	5032		0	0
	31	TEST	5032		0	0
MB_3	32	SEIZE	5152		0	0
	33	ADVANCE	5152		0	0
	34	ASSIGN	5152		0	0
	35	RELEASE	5152		0	0
	36	TEST	5152		0	0
MA_5	37	SEIZE	2566		0	0
	38	ADVANCE	2566		0	0
	39	RELEASE	2566		0	0
MA_7	40	SEIZE	10085		0	0
	41	ADVANCE	10085		0	0
	42	RELEASE	10085		0	0
	43	TRANSFER	10085		0	0
MB_14	44	SEIZE	10000		0	0
	45	ADVANCE	10000		0	0
	46	ASSIGN	10000		0	0

	47	RELEASE	10000	0	0
	48	TEST	10000	0	0
MA_9	49	SEIZE	4997	0	0
	50	ADVANCE	4997	0	0
	51	RELEASE	4997	0	0
	52	TRANSFER	4997	0	0
MB_13	53	SEIZE	5053	0	0
	54	ADVANCE	5053	0	0
	55	ASSIGN	5053	0	0
	56	RELEASE	5053	0	0
	57	TEST	5053	0	0
MA_10	58	TRANSFER	2604	0	0
	59	SEIZE	2504	0	0
	60	ADVANCE	2504	0	0
	61	RELEASE	2504	0	0
	62	TRANSFER	2504	0	0
MB_4	63	SEIZE	4933	0	0
	64	ADVANCE	4933	0	0
	65	ASSIGN	4933	0	0
	66	RELEASE	4933	0	0
	67	TEST	4933	0	0
MA_6	68	SEIZE	4977	0	0
	69	ADVANCE	4977	0	0
	70	RELEASE	4977	0	0
	71	TRANSFER	4977	0	0
MB_8	72	SEIZE	5003	0	0
	73	ADVANCE	5003	0	0
	74	ASSIGN	5003	0	0
	75	RELEASE	5003	0	0
	76	TEST	5003	0	0
	77	TRANSFER	2504	0	0
ME_1	78	SEIZE	10000	0	0
	79	ADVANCE	10000	0	0
	80	RELEASE	10000	0	0
MET	81	TABULATE	10000	0	0
	82	TERMINATE	10000	0	0

FACILITY DELAY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY
MAA_1	10000	0.010	1.000	1	0	0	0	0
0								
YZ1	10000	0.320	32.000	1	0	0	0	0
0								
KAN1	0	0.000	0.000	1	0	0	0	0
0								
YZ2	10000	0.320	32.000	1	0	0	0	0
0								
KAN	10000	0.320	32.000	1	0	0	0	0
0								
MBB_2	20085	0.020	1.000	1	0	0	0	0
0								
MBB_11	10085	0.010	1.000	1	0	0	0	0
0								
MBB_12	5032	0.005	1.000	1	0	0	0	0
0								
MBB_3	5152	0.005	1.000	1	0	0	0	0
0								
MAA_6	4977	0.005	1.000	1	0	0	0	0
0								
MAA_7	10085	0.010	1.000	1	0	0	0	0
0								

MBB_13 0	5053	0.005	1.000	1	0	0	0	0
MBB_4 0	4933	0.005	1.000	1	0	0	0	0
MBB_14 0	10000	0.010	1.000	1	0	0	0	0
MBB_8 0	5003	0.005	1.000	1	0	0	0	0
MAA_10 0	2504	0.003	1.000	1	0	0	0	0
ENDS 0	10000	0.010	1.000	1	0	0	0	0
MAA_5 0	2566	0.003	1.000	1	0	0	0	0
MAA_9 0	4997	0.005	1.000	1	0	0	0	0

TABLE CUM. % TAB	MEAN	STD. DEV.	RANGE		RETRY	FREQUENCY
	43.047	8.131			0	
		36.000	-	37.000		3663
36.63		37.000	-	38.000		1232
48.95		38.000	-	39.000		0
48.95		39.000	-	40.000		0
48.95		40.000	-	41.000		0
48.95		41.000	-	42.000		483
53.78		42.000	-	43.000		1638
70.16		43.000	-	44.000		489
75.05		44.000	-	45.000		0
75.05		45.000	-	46.000		0
75.05		46.000	-	47.000		70
75.75		47.000	-	48.000		346
79.21		48.000	-	49.000		677
85.98		49.000	-	50.000		178
87.76		50.000	-	51.000		0
87.76		51.000	-	52.000		10
87.86		52.000	-	53.000		66
88.52		53.000	-	54.000		219
90.71		54.000	-	55.000		237
93.08		55.000	-	56.000		72
93.80		56.000	-	57.000		2
93.82						

	57.000	–	58.000	13
93.95				
	58.000	–	59.000	53
94.48				
	59.000	–	60.000	112
95.60				
	60.000	–	61.000	106
96.66				
	61.000	–	62.000	21
96.87				
	62.000	–	63.000	1
96.88				
	63.000	–	64.000	11
96.99				
	64.000	–	65.000	31
97.30				
	65.000	–	66.000	61
97.91				
	66.000	–	67.000	39
98.30				
	67.000	–	68.000	13
98.43				
	68.000	–	69.000	6
98.49				
	69.000	–	70.000	6
98.55				
	70.000	–	71.000	21
98.76				
	71.000	–	72.000	31
99.07				
	72.000	–	73.000	7
99.14				
	73.000	–	74.000	2
99.16				
	74.000	–	75.000	1
99.17				
	75.000	–	76.000	5
99.22				
	76.000	–	77.000	11
99.33				
	77.000	–	78.000	7
99.40				
	78.000	–	79.000	11
99.51				
	79.000	–	80.000	2
99.53				
	80.000	–	81.000	1
99.54				
	81.000	–	82.000	9
99.63				
	82.000	–	83.000	5
99.68				
	83.000	–	84.000	7
99.75				
	84.000	–	85.000	2
99.77				
	85.000	–	86.000	0
99.77				
	86.000	–	87.000	1
99.78				
	87.000	–	88.000	4
99.82				

99.85	88.000	-	89.000	3
99.89	89.000	-	90.000	4
99.90	90.000	-	91.000	1
99.90	91.000	-	92.000	0
99.91	92.000	-	93.000	1
99.93	93.000	-	94.000	2
99.95	94.000	-	95.000	2
99.96	95.000	-	96.000	1
99.96	96.000	-	97.000	0
99.96	97.000	-	98.000	0
100.00	98.000	-	-	4

FEC XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE
10001	0	1000100.000	10001	0	1		

Приложение С

Структура компьютерной сети - TCP/IP сеть

