Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К курсовому проектированию

по дисциплине: «Методы моделирования»

на тему: «Имитационное моделирование распределённого приложения для компьютерной сети»

Выполнил:

студент группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Принял:

Зинкин С.А.

Федюнин Р.Н.

Пенза 2024

# Содержание

[Введение 3](#_Toc184507283)

[1 Постановка задачи 4](#_Toc184507284)

[2 Моделирование на Python 5](#_Toc184507285)

[3 Реализация в GPSS WORLD 8](#_Toc184507286)

[4 Моделирование сети Петри в PIPE. 13](#_Toc184507287)

[5 Моделирование сети TCP/IP 16](#_Toc184507288)

[6 Расчёт скорости пересылки данных 17](#_Toc184507289)

[Заключение 18](#_Toc184507290)

[Список литературы 19](#_Toc184507291)

[Приложение А – Листинг на Python 20](#_Toc184507292)

[Приложение B – Листинг на GPSS WORLD 23](#_Toc184507293)

# **Введение**

Моделирование автоматов и систем массового обслуживания играет ключевую роль в современном системном анализе и проектировании, так как позволяет исследовать поведение сложных систем в условиях неопределенности и динамических изменений. В данной работе рассматриваются три основных подхода к моделированию: использование Python для программирования, применение специализированного языка GPSS WORLD для симуляции систем массового обслуживания, а также моделирование с использованием сетей Петри с помощью инструмента PIPE.

Python представляет собой мощный инструмент для создания универсальных моделей, обеспечивая высокую гибкость и возможность адаптации под различные сценарии моделирования. GPSS WORLD же предлагает более формализованный подход к симуляции процессов, что особенно полезно при моделировании многозадачных систем обслуживания. Сети Петри, реализуемые в PIPE, предоставляют удобное средство для анализа процессов синхронизации и решения проблем ресурсных конфликтов в системах с параллельной обработкой.

Цель работы — исследовать особенности каждого из этих подходов и создать сравнительную модель, которая позволит оценить их преимущества и ограничения в различных условиях применения.

# **1 Постановка задачи**

Заданы распределённый алгоритм и граф межузловых связей в компьютерной сети с коммутацией пакетов сообщений. Требуется построить исполнимую или поведенческую имитационную модель передачи сообщений в данной сети с учётом задержек, образования очередей, скорости передачи пакетов по каналам связи. При выборе технических характеристик необходимо ориентироваться на TCP/IP сети и сети Ethernet.

# **2 Моделирование на Python**

В результате выполнения лабораторной работы номер 2, по заданному алгоритму была построена отмеченная ГСА.

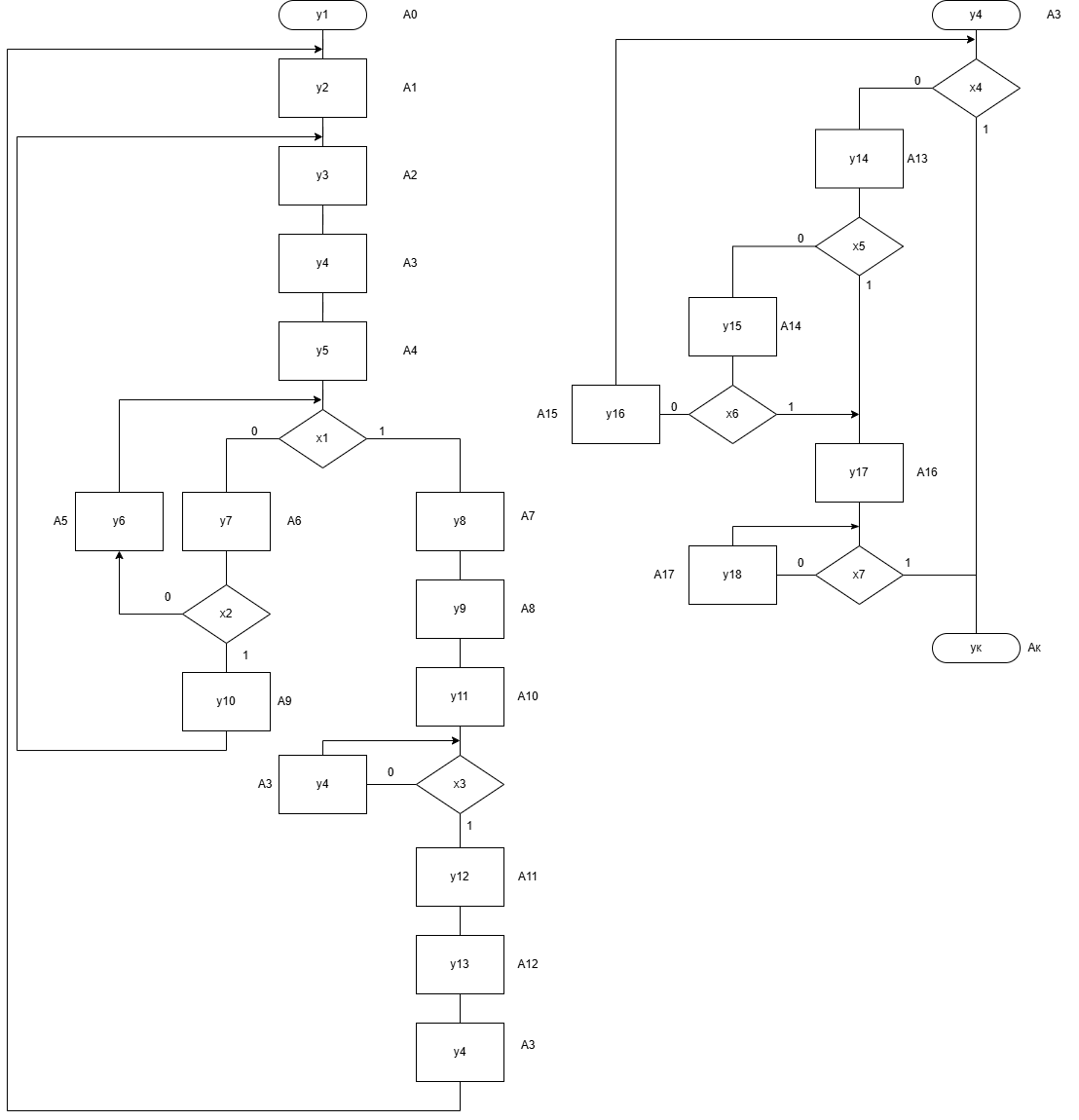


Рисунок 1 – Отмеченная ГСА

Используя полученную ГСА, был построен граф автомата

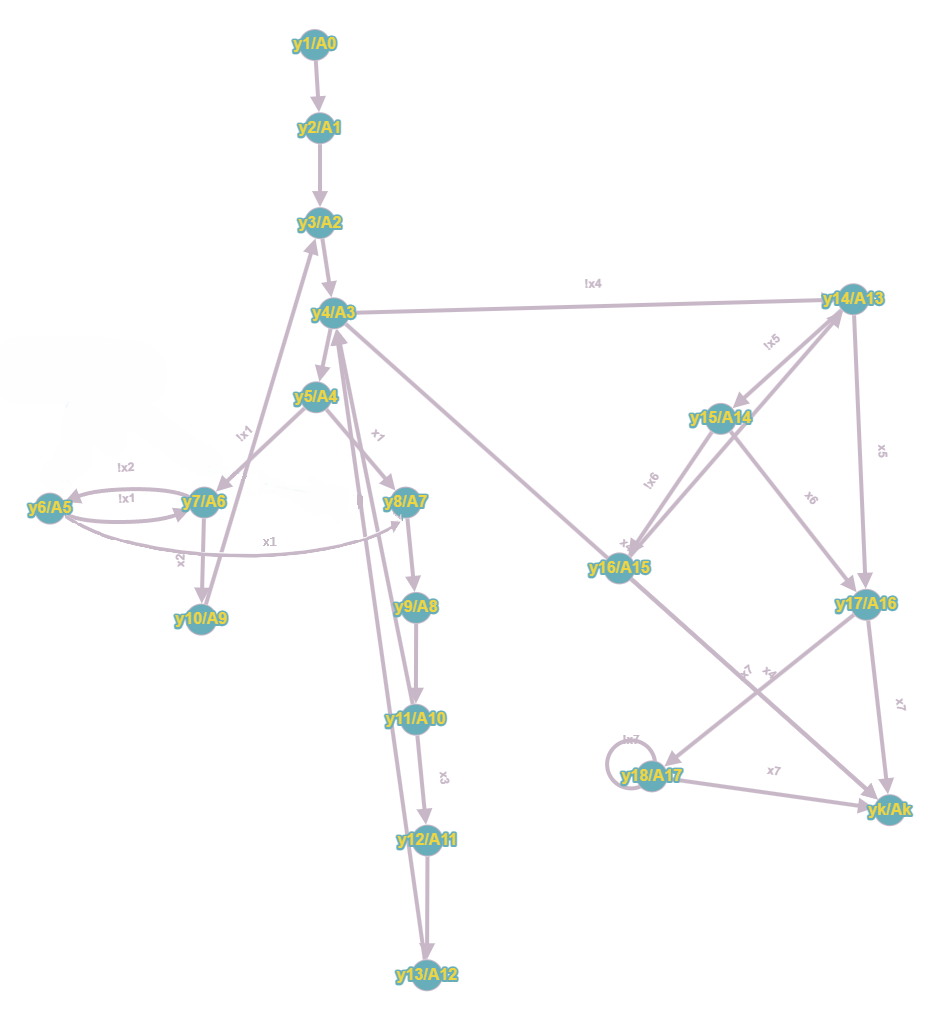


Рисунок 2 – граф автомата

Также промоделировал данный автомат на Python, протестировал на определённых наборах значений. Результат совпал с настоящим.

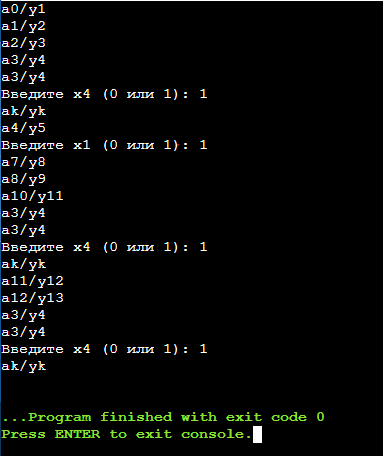


Рисунок 3 – результат работы программы

Листинг программы находится в приложении А

3 Реализация в GPSS WORLD

Используя ГСА алгоритма, промоделировал алгоритм на языке GPSS WORLD и получили статистические данные.

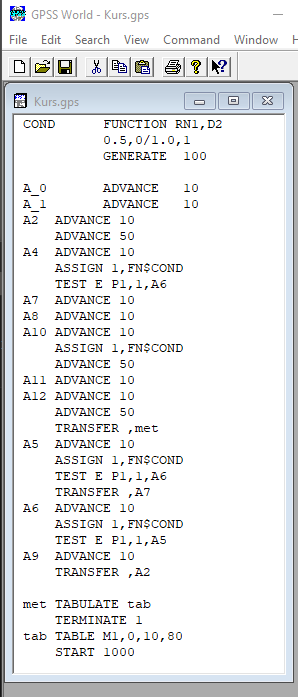


Рисунок 4 – Model файл для GPSS

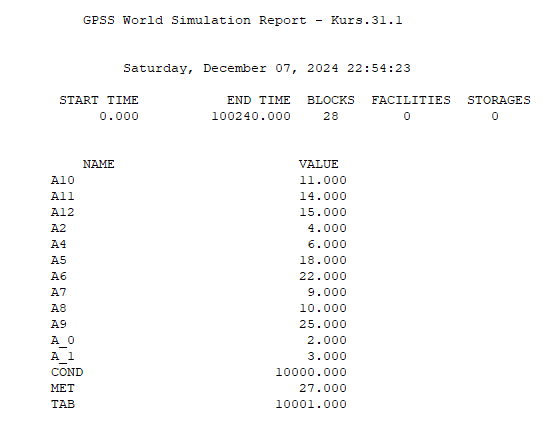


Рисунок 5 – Отчет, часть 1

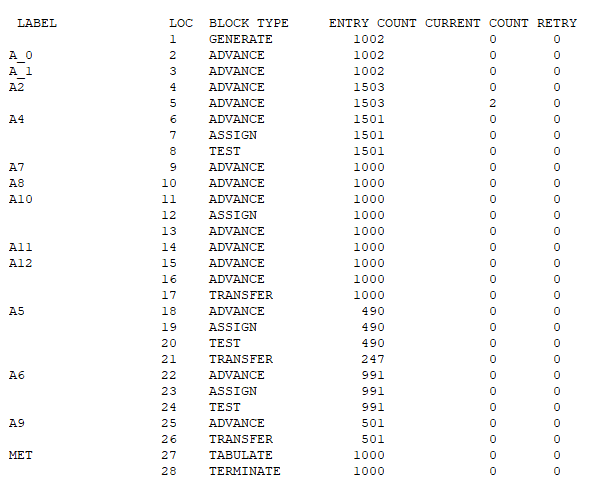


Рисунок 6 – Отчет, часть 2

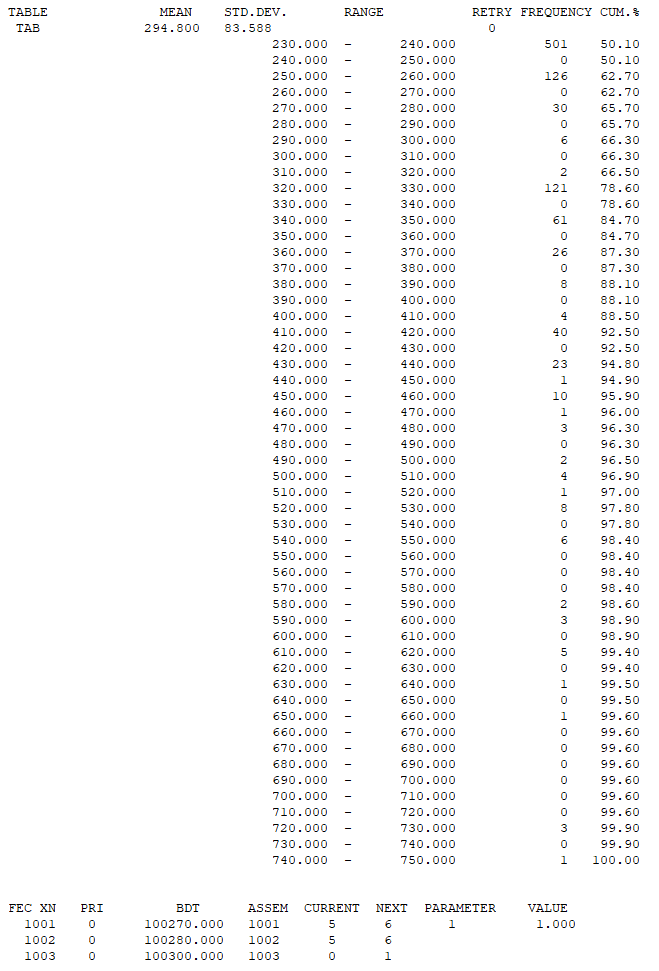


Рисунок 6 – Отчет, часть 3

Далее запустил Simulation Window для просмотра дополнительных статистических данных

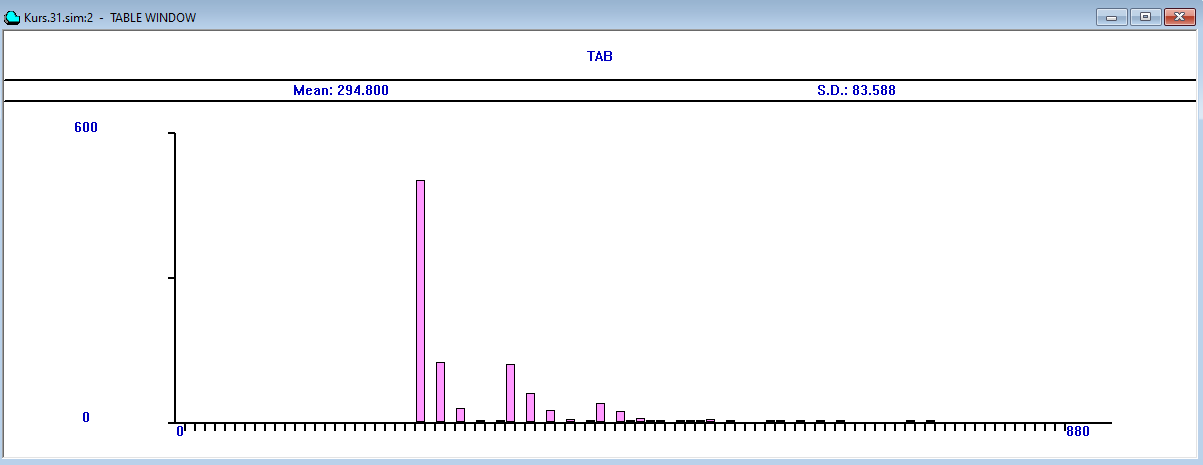


Рисунок 7 – TableWindow

Листинг программы для GPSS WORLD находится в приложении B

Добавил элемент клиент серверного взаимодействия в код GPSS

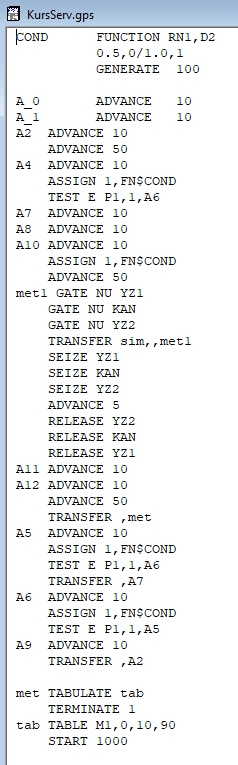


Рисунок 8 – дополненный код на GPSSWORLD

При симуляции получил ожидаемый результат.

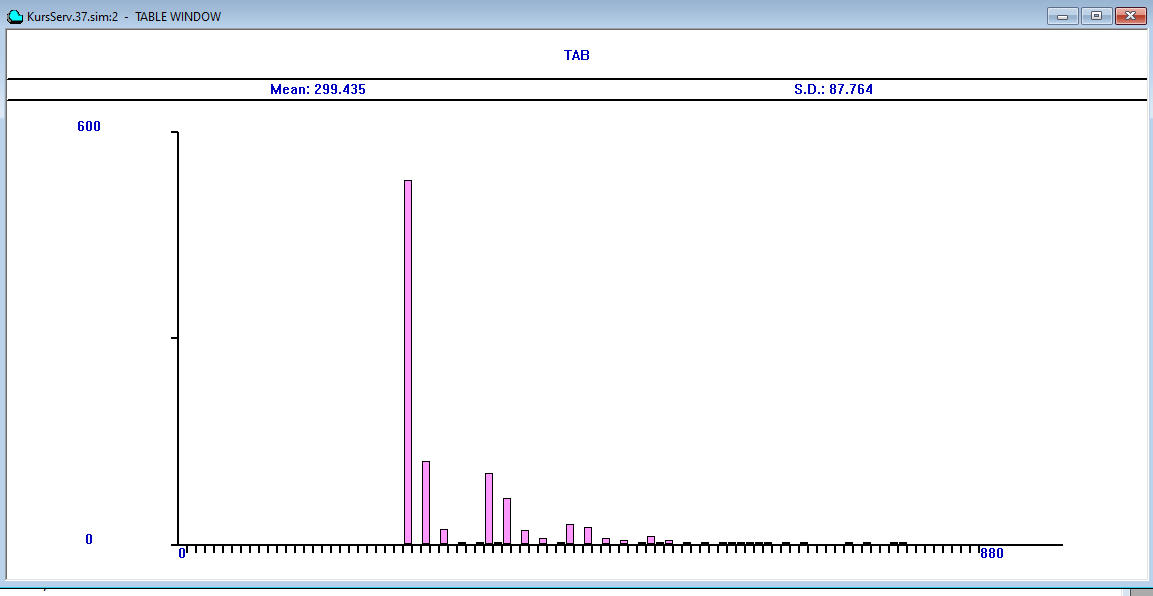


Рисунок 9 – повторная симуляция с помощью Table Window

# **4 Моделирование сети Петри в PIPE.**

Промоделирована в PIPE сеть Петри, построенная на основе алгоритма, который был ранее разработан.

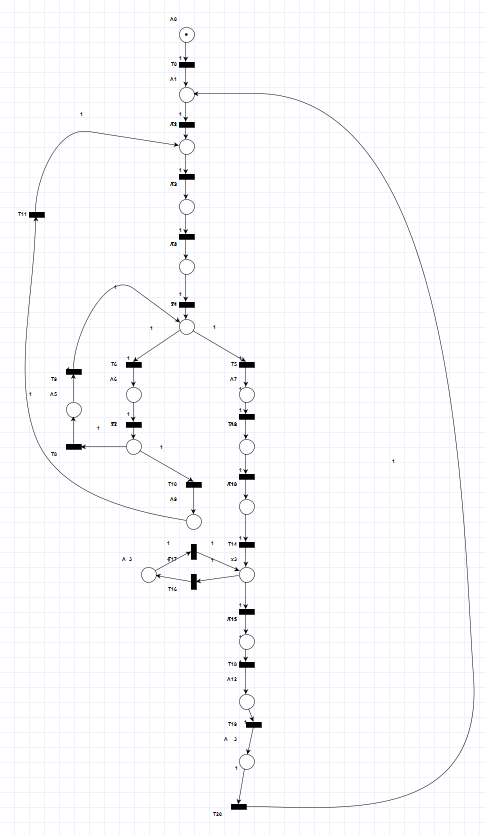


Рисунок 10 – сеть Петри

Проверена работоспособность сети, проведён поиск наличия ошибок, накапливания токенов. В результате сеть работает исправно, ошибок не обнаружено.

Далее был добавлен элемент клиент-серверного взаимодействия

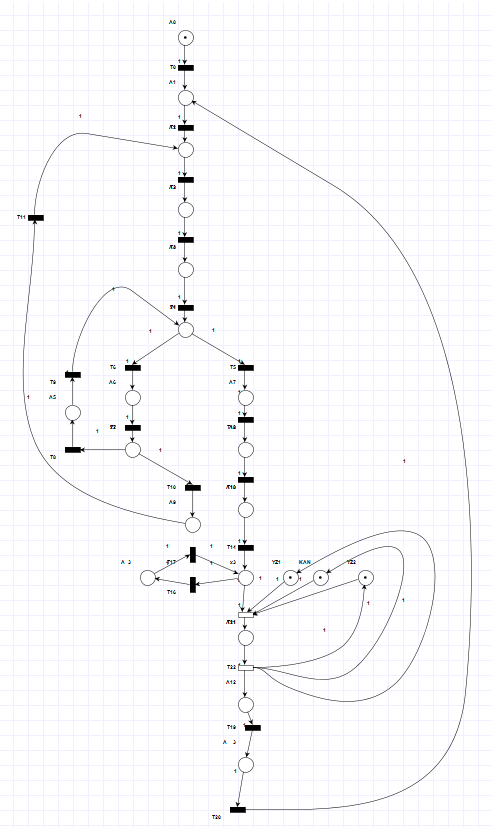


Рисунок 11 – дополненная сеть Петри.

С учётом клиент серверной части промоделировал сеть Петри, алгоритм работает исправно.

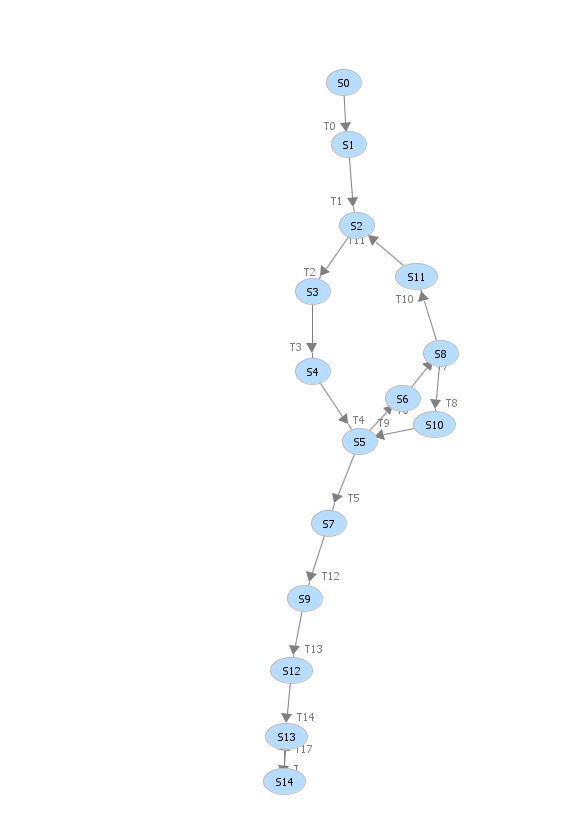
Сгенерировал граф достижимости

Рисунок 12 – граф достижимости

# **5 Моделирование сети TCP/IP**

С помощью симулятора сети передачи данных Cisco Packet Tracer была создана модель TCP/IP сети, демонстрирующая взаимодействие клиента и сервера

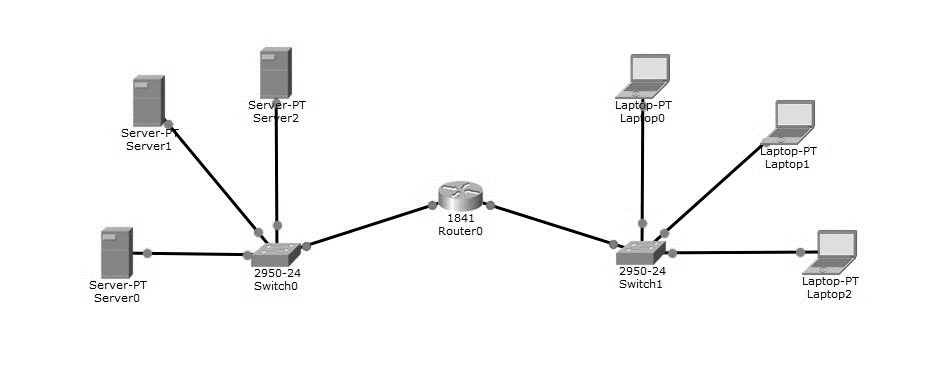


Рисунок 13 – TCP/IP сеть

# **6 Расчёт скорости пересылки данных**

В рамках курсовой работы были разработаны несколько имитационных моделей для анализа времени, необходимого для передачи файла. В ходе исследования были проведены различные расчёты.

В реальных условиях на скорость передачи данных могут оказывать влияние следующие факторы:  
V – скорость передачи файлов = 2 Мбит/с = 2 097 152 бит/с  
I – размер файла = 64 КБ = 524 288 бит  
t = I / V = 524 288 / 2 097 152 = 0.25 c  
 В реальной схеме на скорость передачи данных могут также повлиять следующие факторы:  
1. **Потеря пакетов и восстановление**: В случае потери пакетов время передачи может увеличиться, так как данные необходимо передавать повторно, что ведет к дополнительным затратам времени на восстановление связи..  
2. **Протоколы передачи данных**: Различные протоколы, такие как TCP или UDP, могут существенно влиять на скорость передачи из-за особенностей управления потоком и методов коррекции ошибок.  
3. **Задержки в сети**: Помимо скорости передачи, важно учитывать задержки, возникающие в процессе передачи данных, которые могут быть вызваны различными факторами, такими как расстояние, нагрузка на сеть и технические характеристики оборудования.

Эти факторы могут значительно повлиять на общий процесс передачи данных в реальных условиях сети.

# **Заключение**

В процессе выполнения курсового проекта были исследованы различные методы моделирования, такие как Python, GPSS WORLD и сети Петри в PIPE. Каждый из этих подходов имеет свои особенности, которые делают его подходящим для определённых типов задач: Python предоставляет большую гибкость в настройке моделей и их адаптации к конкретным условиям, GPSS WORLD позволяет точно моделировать процессы массового обслуживания и учитывать временные задержки, а сети Петри эффективно применяются для моделирования параллельных процессов и анализа синхронизации.

Результаты моделирования помогли выявить ключевые характеристики сети и оценить её поведение в разных условиях эксплуатации. Заключение исследования подтвердило, что использование данных методов моделирования является эффективным инструментом для анализа и оптимизации работы распределённых систем.

# **Список литературы**

1 Козлов А.В., Белов Н.И. Основы моделирования систем с дискретными событиями. Учебное пособие. – М.: Физматлит, 2021.

2 Шишкин А.В. Симуляция и моделирование систем массового обслуживания. – СПб.: БХВ-Петербург, 2020.

3 Левин Г.Я. Введение в теорию автоматов и вычислительных систем. – М.: ДМК Пресс, 2019.

4 Смирнов И.А. Моделирование процессов с использованием Python. – М.: Наука, 2022.

5 Глушков В.П. Основы симуляции и моделирования: от теории к практике. – М.: ИТД, 2021.

6 Петров А.П. Моделирование и анализ сложных систем: практическое руководство. – Новосибирск: Сибирское университетское издательство, 2020.

# **Приложение А**

(Обязательное)

Листинг на Python

a = 0

x = 0

def podprogram():

global a

print("a3/y4")

while(1):

x = int(input("Введите х4 (0 или 1): "))

if(x == 0):

print("a13/y14")

a = 13

elif(x == 1):

print("ak/yk")

return

elif(a == 13):

x = int(input("Введите х5 (0 или 1): "))

if(x == 0):

print("a14/y15")

a = 14

x = int(input("Введите х6 (0 или 1): "))

if(x == 0):

print("a15/y16")

a = 15

print("a13/y14")

a = 13

else:

print("a16/y17")

a = 16

else:

print("a16/y17")

a = 16

x = int(input("Введите х7 (0 или 1): "))

if(x == 0):

a = 17

print("a17/y18")

x = int(input("Введите х7 (0 или 1): "))

if(x == 0):

a = 17

print("a17/y18")

else:

return

else:

return

while(1):

if(a == 3):

x = int(input("Введите х4 (0 или 1): "))

if(x == 0):

print("a13/y14")

a = 13

else:

print("ak/yk")

return

if(a == 13):

x = int(input("Введите х5 (0 или 1): "))

if(x == 0):

print("a14/y15")

a = 14

x = int(input("Введите х6 (0 или 1): "))

if(x == 0):

print("a15/y16")

a = 15

print("a13/y14")

a = 13

else:

print("a16/y17")

a = 16

else:

print("a16/y17")

a = 16

x = int(input("Введите х7 (0 или 1): "))

if(x == 0):

a = 17

print("a17/y18")

x = int(input("Введите х7 (0 или 1): "))

if(x == 0):

a = 17

print("a17/y18")

else:

return

else:

return

while(1):

if(a == 0):

print("a0/y1")

a = 1

elif(a == 1):

print("a1/y2")

a = 2

elif(a == 2):

print("a2/y3")

a = 3

elif(a == 3):

print("a3/y4")

podprogram()

print("a4/y5")

a = 4

elif(a == 4):

x = int(input("Введите х1 (0 или 1): "))

if(x == 0):

a = 6

print("a6/y7")

else:

a = 7

print("a7/y8")

elif(a == 6):

x = int(input("Введите х2 (0 или 1): "))

if(x == 0):

a = 5

print("a5/y6")

else:

a = 9

print("a9/y10")

elif(a == 5):

x = int(input("Введите х1 (0 или 1): "))

if(x == 0):

a = 6

print("a6/y7")

else:

a = 7

print("a7/y8")

elif(a == 9):

a = 2

print("a2/y3")

elif(a == 7):

a = 8

elif(a == 8):

a = 10

print("a8/y9")

elif(a == 10):

print("a10/y11")

print("a3/y4")

podprogram()

a = 11

elif(a == 11):

print("a11/y12")

a = 12

elif(a == 12):

print("a12/y13")

print("a3/y4")

podprogram()

break

# **Приложение B**

(Обязательное)

Листинг на GPSS WORLD

COND FUNCTION RN1,D2

0.5,0/1.0,1

GENERATE 100

A\_0 ADVANCE 10

A\_1 ADVANCE 10

A2 ADVANCE 10

ADVANCE 50

A4 ADVANCE 10

ASSIGN 1,FN$COND

TEST E P1,1,A6

A7 ADVANCE 10

A8 ADVANCE 10

A10 ADVANCE 10

ASSIGN 1,FN$COND

ADVANCE 50

A11 ADVANCE 10

A12 ADVANCE 10

ADVANCE 50

TRANSFER ,met

A5 ADVANCE 10

ASSIGN 1,FN$COND

TEST E P1,1,A6

TRANSFER ,A7

A6 ADVANCE 10

ASSIGN 1,FN$COND

TEST E P1,1,A5

A9 ADVANCE 10

TRANSFER ,A2

met TABULATE tab

TERMINATE 1

tab TABLE M1,0,10,90

START 1000