Санкт-Петербургский политехнический университет Петра Великого

Физико-механический институт

Высшая школа прикладной математики и вычислительной физики

Компьютерные сети

Отчет на тему: "Симуляция сети с использованием протоколов GBN и OSPF"

Выполнил:

Студент: Файзрахманов Артур Рафаэлевич
Группа: 5040102/30201

Принял:

к. ф.-м. н.
Баженов Александр Николаевич

Санкт-Петербург
2025 г.

Содержание

1 Введение

В данном проекте реализована симуляция сети с использованием протоколов Go-Back-N (GBN) и Open Shortest Path First (OSPF). Основная цель проекта — изучение взаимодействия узлов в сети, передачи данных и визуализация ключевых процессов, таких как маршрутизация, передача файлов по частям и обработка разрывов соединений.

2 Описание функциональности

2.1 Симуляция сети

Сеть состоит из нескольких узлов, соединенных двунаправленными каналами связи. Визуализация выполнена с использованием библиотеки `tkinter`, что позволяет интерактивно управлять состоянием соединений и наблюдать за процессом передачи данных.

2.2 Реализованные возможности

1. **Маршрутизация с использованием OSPF.** Каждый узел рассчитывает кратчайшие пути до других узлов с помощью алгоритма Дейкстры. Таблицы маршрутизации автоматически обновляются при разрыве или восстановлении соединений.

2. **Передача данных с использованием GBN.** Реализован протокол Go-Back-N, обеспечивающий передачу пакетов с подтверждением (ACK). В случае потери пакета происходит повторная отправка.

3. **Передача файлов по частям.** Данные (файлы) хранятся в распределенном хранилище. Узел-запросчик получает файл, собирая его из кусочков, передаваемых от нескольких узлов одновременно.

4. **Разрыв и восстановление соединений.** Пользователь может разрывать соединения между узлами. При этом таблицы маршрутизации пересчитываются в реальном времени.

5. **Визуализация передачи данных.** Каждый пакет визуализируется на экране как движущийся объект. ACK и повторная передача также отображаются.

6. **Потеря пакетов.** Для проверки устойчивости GBN введена случайная потеря пакетов с заданной вероятностью.

3 Архитектура проекта

3.1 Общая структура

Проект построен на модульной архитектуре, где каждый компонент отвечает за определенную часть функциональности. Основные модули взаимодействуют через четко определенные интерфейсы, что упрощает понимание, тестирование и расширение системы.

3.2 Основные компоненты

3.2.1 Узлы сети

Каждый узел представляет собой самостоятельную сущность в сети. Узлы обладают следующими характеристиками:

- **ID узла:** Уникальный идентификатор, используемый для адресации.
- **Таблица маршрутизации:** Содержит кратчайшие пути до всех других узлов. Расчитывается на основе OSPF.
- **Локальное хранилище данных:** Узел может хранить файлы, представленные в виде набора кусков данных.
- **Функции передачи данных:** Узел может отправлять и получать пакеты с использованием протокола GBN.

3.2.2 Соединения

Соединения реализованы как двунаправленные каналы связи между узлами. Основные свойства:

- **Состояние соединения:** Активное или разорванное.
- **Стоимость:** Метрика, используемая в алгоритме маршрутизации OSPF.
- **Потеря пакетов:** Случайная потеря пакетов с заданной вероятностью для тестирования устойчивости GBN.

3.2.3 Протоколы

Проект включает два ключевых протокола:

1. **OSPF:** Используется для расчета таблиц маршрутизации. Каждый узел рассылает сообщения о состоянии соединений, на основе которых строится глобальная карта сети.
2. **GBN:** Обеспечивает передачу пакетов между узлами. Использует механизм подтверждений (ACK) и повторной отправки в случае потерь.

3.2.4 Пользовательский интерфейс

Интерфейс построен на основе библиотеки `tkinter`. Основные элементы:

- **Визуализация сети:** Узлы представлены как круги, соединения между ними — как линии.
- **Интерактивность:** Пользователь может разрывать соединения, запускать передачу файлов и наблюдать за процессами в реальном времени.
- **Отображение данных:** В процессе передачи визуализируются пакеты, подтверждения (ACK) и повторные отправки.

3.3 Потоки выполнения

Проект использует многопоточность для разделения задач:

1. **Основной поток:** Обрабатывает интерфейс пользователя, поддерживая его отзывчивость.
2. **Потоки передачи данных:** Каждый запрос на передачу файла или куска данных выполняется в отдельном потоке.
3. **Потоки маршрутизации:** Узлы обмениваются таблицами маршрутизации в фоновом режиме.

Взаимодействие компонентов

1. **Инициализация сети:**
   - Пользователь задает начальную топологию сети.
   - Узлы обмениваются информацией о соединениях для расчета таблиц маршрутизации.
2. **Запрос данных:**
   - Узел-запросчик отправляет запрос на получение файла.
   - Сеть определяет узлы, на которых хранятся куски файла.
   - Куски передаются параллельно, с учетом маршрутизации.
3. **Разрыв соединений:**
   - Пользователь разрывает соединение между двумя узлами.
   - Узлы обновляют свои таблицы маршрутизации.
   - Передача данных перенаправляется через альтернативные маршруты.

3.4 Пример работы

Рассмотрим сценарий передачи файла:

1. Узел `A` запрашивает файл `data.txt`.
2. Файл разбит на 3 части и хранится на узлах `B`, `C` и `D`.
3. Узлы `B`, `C` и `D` отправляют куски файла узлу `A`.
4. Если на пути теряется пакет, GBN инициирует повторную передачу.
5. Узел `A` собирает файл и уведомляет пользователя о завершении.

4 Алгоритмы

4.1 Go-Back-N (GBN)

Протокол Go-Back-N (GBN) относится к категории протоколов передачи данных с подтверждением (ACK) и скользящим окном. Его ключевая цель — гарантированная доставка данных при возможной потере пакетов.

Основные элементы протокола:

1. **Скользящее окно:** В каждый момент времени передающий узел может отправить несколько пакетов (до размера окна). Эти пакеты находятся в состоянии ожидания подтверждения.

2. **Подтверждения (ACK):** Получатель отправляет подтверждение при успешной доставке пакета. Если пакет теряется или повреждается, подтверждение не отправляется.

3. **Повторная передача:** Если подтверждение не приходит в течение тайм-аута, передающий узел повторно отправляет все пакеты, начиная с потерянного.

4.2 Open Shortest Path First (OSPF)

OSPF — это протокол маршрутизации, основанный на алгоритме состояния каналов (link-state). Он широко используется для динамического определения маршрутов в IP-сетях.

Основные этапы работы OSPF:

1. **Изучение топологии:** Каждый узел в сети изучает подключенные к нему узлы и формирует сообщение о состоянии своих соединений (LSA — Link State Advertisement). Узлы обмениваются LSA, чтобы получить полную информацию о топологии сети.

2. **Построение графа:** После получения информации о состоянии всех каналов узел строит граф сети, где вершины — это узлы, а ребра — соединения между ними с указанной стоимостью (метрикой).

3. **Алгоритм Дейкстры:** Каждый узел запускает алгоритм Дейкстры для расчета кратчайших путей от себя до всех остальных узлов. Результат работы алгоритма — таблица маршрутизации, содержащая для каждого назначения лучший следующий узел (next-hop).

Алгоритм Дейкстры:

1. Инициализировать расстояния до всех узлов как бесконечные, кроме исходного узла (расстояние до него равно 0).

2. Добавить исходный узел в множество посещенных узлов.

3. Для каждого непосещенного соседа текущего узла пересчитать расстояния:

$$d[v] = \min(d[v], d[u] + w(u, v)),$$

где $d[v]$ — расстояние до узла v, u — текущий узел, $w(u, v)$ — стоимость ребра между u и v.

4. Выбрать следующий узел с минимальным расстоянием и повторить процесс, пока все узлы не будут посещены.

Реакция на изменения в сети: При разрыве или добавлении соединений узлы пересылают обновленные LSA, после чего все узлы пересчитывают свои таблицы маршрутизации.

5 Результаты

5.1 Тестовые сценарии

1. Запрос файла с нескольких узлов. Убедиться, что файл собирается корректно.

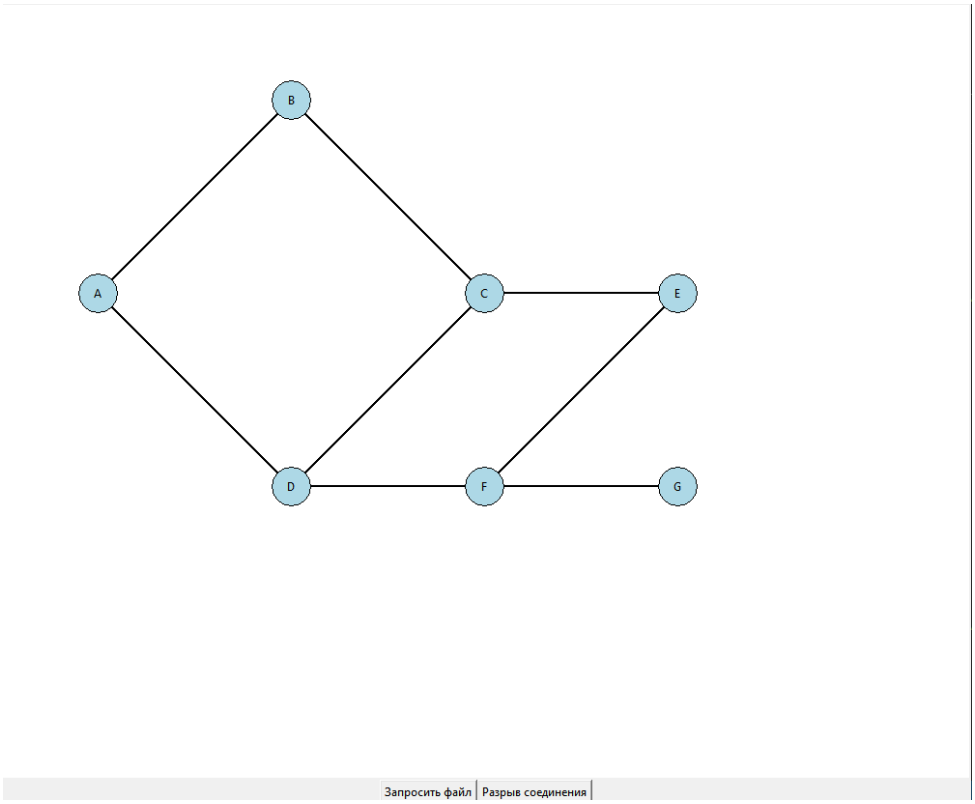

Рис. 1: Интерфейс приложения

2. Разрыв соединения между двумя узлами. Проверить, что передача данных выполняется по альтернативным маршрутам.

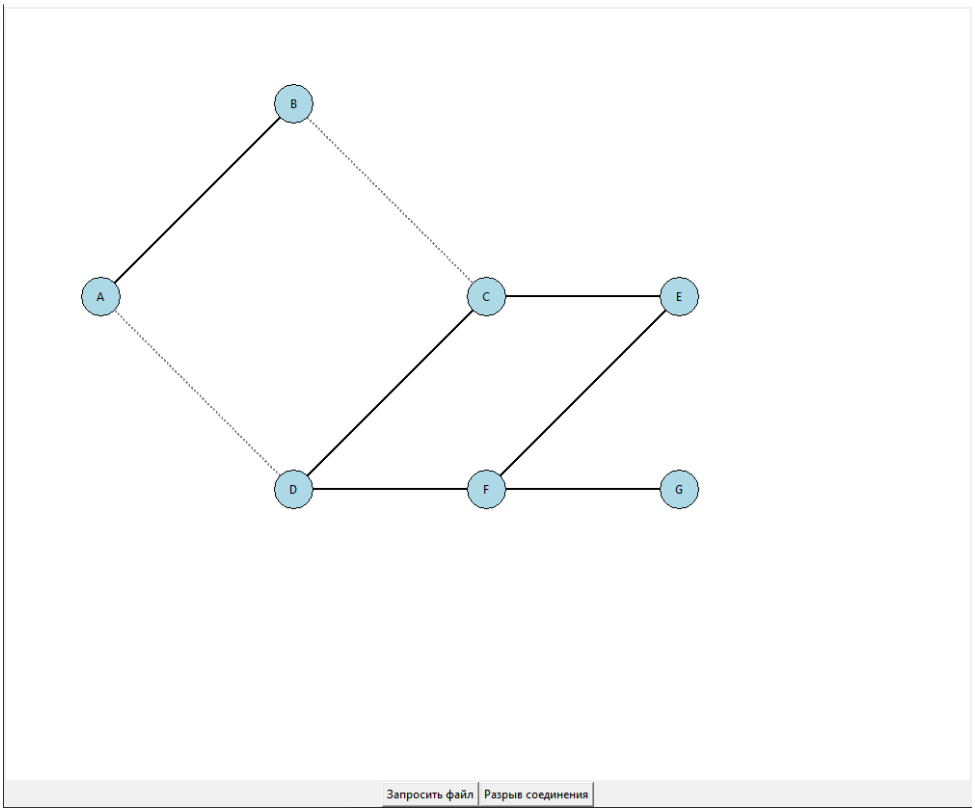

Рис. 2: Отображение топологии с разрывами

3. Установка высокой вероятности потери пакетов. Проверить, что протокол GBN корректно обрабатывает повторные передачи.

```
Распределение данных по узлам:
Узел A: {'file1': ['chunk1'], 'file2': ['chunk1']}
Узел B: {'file1': ['chunk2'], 'file2': ['chunk2']}
Узел C: {'file1': ['chunk3']}
Узел D: {}
Узел E: {}
Узел F: {}
Узел G: {}
Соединение между B и C разорвано.
Соединение между A и D разорвано.
A запрашивает файл file1 у сети.
Нет маршрута из C в A
Файл file1 собран частично на A: ['chunk1', 'chunk2']
Соединение между B и C восстановлено.
Соединение между A и D восстановлено.
A запрашивает файл file1 у сети.
Файл file1 собран частично на A: ['chunk1', 'chunk2', 'chunk3']
```

Рис. 3: Пример работы приложения с выводом в консоль

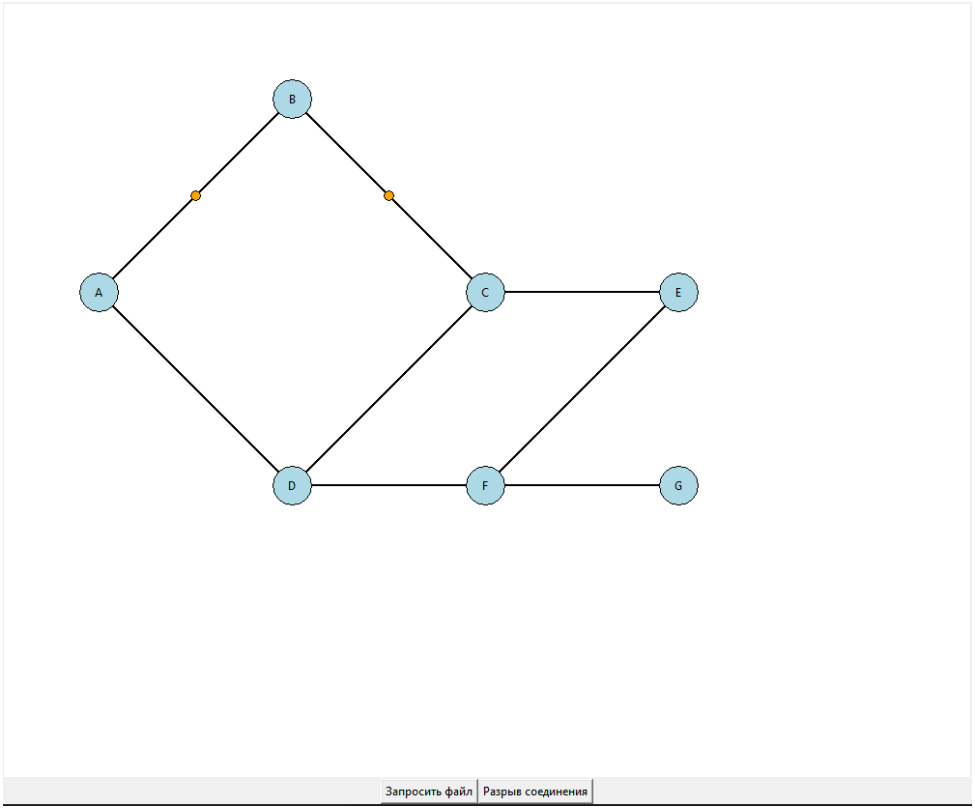

Рис. 4: Пример передачи кусков файла в узел A

6 Заключение

Реализованная симуляция позволяет изучить работу протоколов GBN и OSPF в сети. Благодаря визуализации и интерактивности пользователи могут наблюдать за процессами передачи данных, изменения топологии и восстановления маршрутов. В будущем возможно расширение функциональности, включая реализацию дополнительных протоколов и сложных сценариев взаимодействия узлов.

7 Приложение

7.1 Код программы

Код программы доступен в репозитории:
https://github.com/Hembos/computerNetworks/tree/main/lab3