Отчёт по лабораторной №2

«Реализация протокола динамической маршрутизации OSPF»

по дисциплине

«Компьютерные сети»

Выполнила: Кацман Н.И.

группа: 3640102/90201

Проверил: к.ф.-м.н., доцент

Баженов А.Н.

Оглавление

[Постановка задачи 3](#_Toc67867016)

[Теоретическая часть 3](#_Toc67867017)

[Протокол Open Shortest Path First (OSPF) 3](#_Toc67867018)

[Краткое описание работы протокола 3](#_Toc67867019)

[Топологии сетей 4](#_Toc67867020)

[Топология Bus 4](#_Toc67867021)

[Топология Ring 4](#_Toc67867022)

[Топология Star 5](#_Toc67867023)

[Топология Mesh 5](#_Toc67867024)

[Практическая часть 5](#_Toc67867025)

[Общие положения 5](#_Toc67867026)

[Топология Bus 5](#_Toc67867027)

[Организация экспериментов 5](#_Toc67867028)

[Пример работы протокола 6](#_Toc67867029)

[Топология Ring 6](#_Toc67867030)

[Организация экспериментов 6](#_Toc67867031)

[Пример работы протокола 7](#_Toc67867032)

[Топология Star 7](#_Toc67867033)

[Организация экспериментов 7](#_Toc67867034)

[Пример работы протокола 8](#_Toc67867035)

[Топология Mesh 8](#_Toc67867036)

[Организация экспериментов 8](#_Toc67867037)

[Пример работы протокола 8](#_Toc67867038)

[Заключение 8](#_Toc67867039)

[Список литературы 9](#_Toc67867040)

# Постановка задачи

Реализовать на выбранном высокоуровневом языке протокол динамической маршрутизации OSPF и провести серию симуляций его работы.

# Теоретическая часть

## Протокол Open Shortest Path First (OSPF)

OSPF (Open Shortest Path First) — протокол динамической маршрутизации, основанный на технологии отслеживания состояния канала (link-state technology) и использующий для нахождения кратчайшего пути алгоритм Дейкстры.

## Краткое описание работы протокола

1. Включить OSPF на маршрутизаторе
2. Маршрутизатор выбирает Router ID (уникальное имя маршрутизатора)
3. Включить OSPF на интерфейсах (чтобы протокол знал о каких интерфейсах можно сообщать другим маршрутиазторам)
4. Обнаружение соседей с помощью Hello-пакетов
   1. Маршрутизаторы обмениваются hello-пакетами через все интерфейсы, на которых активирован OSPF.
   2. Маршрутизаторы, которые находятся в одном широковещательном сегменте, становятся соседями, когда они приходят к договоренности об определенных параметрах, указанных в их hello-пакетах.
5. Adjacency (отношения соседства, отношения смежности) это тип соседства между маршрутизаторами, по которому они синхронизируют LSDB. Установка этих отношений зависит от типа сети:
   1. Если маршрутизаторы находятся в сети с множественным доступом, они выбирают DR и выполняют синхронизацию LSDB с ним
   2. Если маршрутизаторы находятся в сети point-to-point, они приступают к синхронизации LSDB друг с другом
6. Синхронизация LSDB. Происходит в несколько этапов. По сформированным отношениям соседства происходит обмен такими пакетами:
   1. DBD (краткое описание LSA в LSDB). С помощью этих пакетов маршрутизаторы сообщают друг другу о том, какую информацию они знают, в сокращенном виде
   2. LSR. После обмена DBD-пакетами, с помощью LSR маршрутизаторы запрашивают у соседа недостающую информацию
   3. LSU (содержит полное описание LSA). В ответ на LSR, который ему прислал сосед, маршрутизатор отправляет LSU, с полным описанием информации, которой не хватает у соседа
   4. LSAck. После получения LSU от соседа, маршрутизатор отправляет подтверждение, что он получил информацию
   5. Если оба маршрутизатора должны запросить друг у друга информацию, то эта процедура повторяется и в другую сторону.
   6. После этого, LSDB синхронизирована, а значит, полностью одинакова между соседями
7. После синхронизации LSDB, маршрутизатор отправляет обновление далее, своим соседям в других широковещательных сегментах
8. Рассылая объявления через зону, все маршрутизаторы строят идентичную LSDB
9. Когда база данных построена, каждый маршрутизатор использует алгоритм SPF (shortest path first) для вычисления графа без петель, который будет описывать кратчайший путь к каждому известному пункту назначения с собой в качестве корня. Этот граф — дерево кратчайшего пути.
10. Каждый маршрутизатор строит таблицу маршрутизации, основываясь на своем дереве кратчайшего пути.

## Топологии сетей

Под топологией сети понимается описание ее физического расположения, а именно, как компьютеры соединены друг с другом в сети, и с помощью каких устройств входят в физическую топологию. Существуют четыре основных топологии: шина (Bus), кольцо (Ring), звезда (Star) и ячеистая топология (Mesh). Другие топологии обычно являются комбинацией двух и более главных типов.

### Топология Bus

Физическая топология "шина" (Bus), именуемая также линейной шиной (Linear Bus), состоит из единственного кабеля, к которому присоединены все компьютеры сегмента

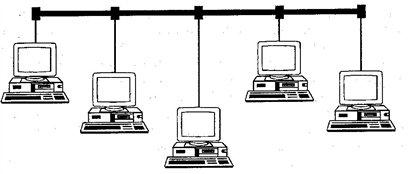


Рисунок 1 Топология Bus

### Топология Ring

Топология "кольцо" (Ring) обычно используется в сетях Token Ring и FDDI (волоконно-оптических). В физической топологии Ring линия передачи данных фактически образует логическое кольцо, к которому подключены все компьютеры сети.

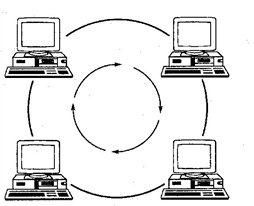


Рисунок 2 Топология Ring

### Топология Star

В топологии "звезда" (Star) все компьютеры в сети соединены друг с другом с помощью центрального концентратора.

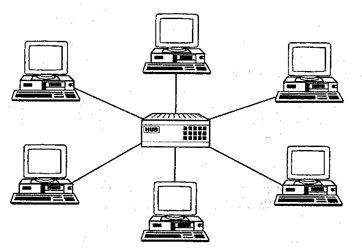


Рисунок 3 Топология Star

### Топология Mesh

Ячеистая топология (Mesh) соединяет все компьютеры попарно.

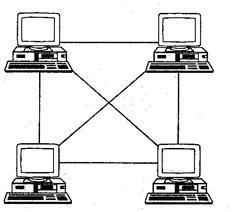


Рисунок 4 Топология Mesh

# Практическая часть

## Общие положения

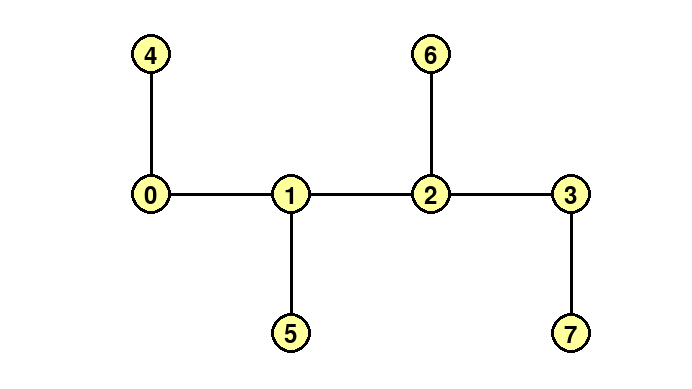
Для выполнения данной работы был выбран язык **Python 3.5.**

В ходе работы были проведены симуляции работы протокола OSPF для четырёх основных типов топологии сетей: Bus, Ring, Star и Mesh

## Топология Bus

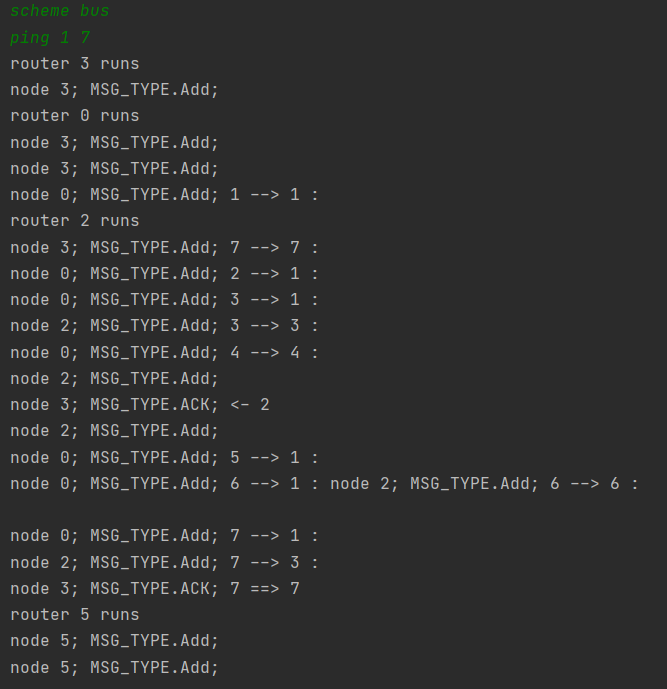
### Организация экспериментов

Была смоделирована сеть с топологией Bus из 8-ми узлов. Её схематическое изображение представлено на рисунке ниже.



### Пример работы протокола

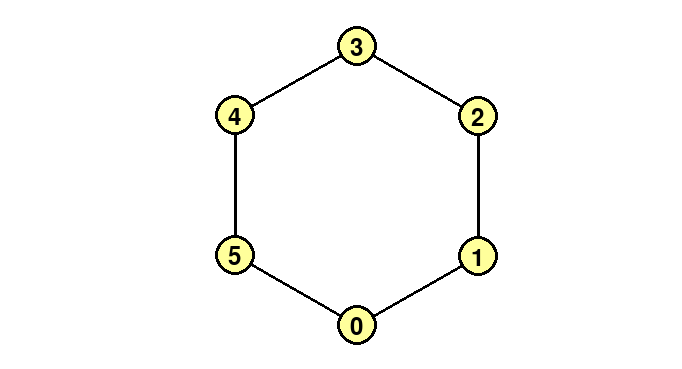
На сети схематично изображённой выше был запущен протокол OSPF для коннекта между первым и седьмым узлами. Ниже приведён работы протокола OSPF для этой схемы.



## Топология Ring

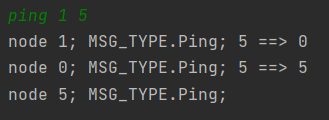
### Организация экспериментов

Была смоделирована сеть с топологией Ring из 6-ми узлов. Её схематическое изображение представлено на рисунке ниже.



### Пример работы протокола

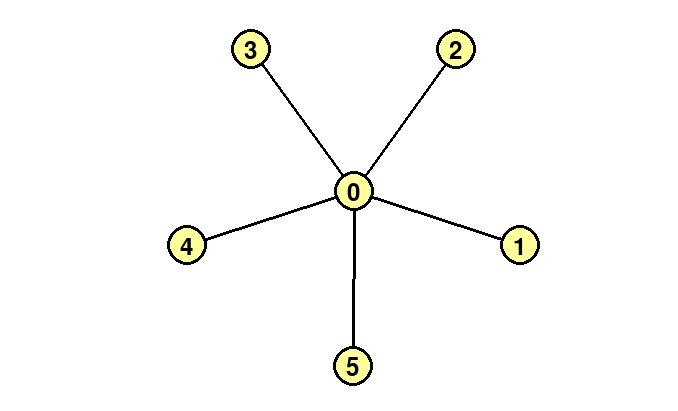
На сети схематично изображённой выше был запущен протокол OSPF для коннекта между первым и пятым узлами. Ниже приведён маршрут по которому между этими узлами передавались бы данные при работе протокола OSPF.



## Топология Star

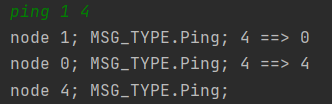
### Организация экспериментов

Была смоделирована сеть с топологией Star из 6-ми узлов. Её схематическое изображение представлено на рисунке ниже.



### Пример работы протокола

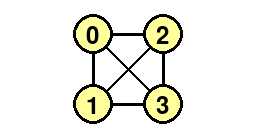
На сети схематично изображённой выше был запущен протокол OSPF для коннекта между первым и четвёртым узлами. Ниже приведён маршрут по которому между этими узлами передавались бы данные при работе протокола OSPF.



## Топология Mesh

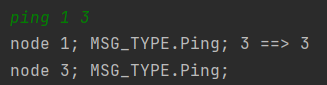
### Организация экспериментов

Была смоделирована сеть с топологией Mesh из 4-х узлов. Её схематическое изображение представлено на рисунке ниже.



### Пример работы протокола

На сети схематично изображённой выше был запущен протокол OSPF для коннекта между первым и третьим узлами. Ниже приведён маршрут по которому между этими узлами передавались бы данные при работе протокола OSPF.



# Заключение

В ходе работы был реализован протокол динамической маршрутизации OSPF и проведены симуляции его работы для четырёх основных типов топологии сетей.

Для всех четырёх топологий маршруты между запрошенными узлами были найдены и действительно являлись оптимальными маршрутами между этими узлами.

Файлы кода и этот отчёт можно посмотреть [здесь](https://github.com/Hadegda/NetworkOSPFProtocol).

# Список литературы

1. Протокол OSPF [Электронный ресурс]   
   url: http://xgu.ru/wiki/OSPF  
   Дата обращения: 28.03.21
2. Протокол OSPF [Электронный ресурс]   
   url: https://habr.com/ru/post/418391/  
   Дата обращения: 28.03.21
3. Топологии сетей [Электронный ресурс]   
   url: http://mydirectx.ru/seti/topologiia\_seti\_i\_ee\_vidy.htm#:~:text= Существуют%20четыре%20основных%20топологии%3A%20шина,из%20первых%20шагов%20планирования%20сети.   
   Дата обращения: 28.03.21