Санкт-Петербургский политехнический университет
Петра Великого

Физико-механический институт

Отчёт по лабораторной работе на тему:
Реализация протокола маршрутизации Open Shortest Path First

Выполнил студент:
Завьялов В. В.
группа: 5040102/30201

Проверил:
к.ф.-м.н., доцент
Баженов А. Н.

Санкт-Петербург
2024 г.

Содержание

1 Постановка задачи

Необходимо реализовать протокол маршрутизации OSPF (Open Shortest Path First) и проверить его работоспособность на следующих видах топологий сети: линейной, кольцевидной и звёздной.

2 Теория

Протокол OSPF (Open Shortest Path First) является одним из наиболее широко применяемых протоколов внутренней маршрутизации (IGP) в современных IP-сетях. Он основан на принципе протокола состояния каналов (Link State Protocol), предполагающем обмен информацией о текущем состоянии всех каналов между маршрутизаторами внутри одной автономной системы. В отличие от протоколов, ориентированных на дистанционно-векторный подход, OSPF поддерживает точное знание глобальной структуры сети, а не ограничивается информацией о соседях или кратчайших дистанциях до определённых подсетей.

Основой работы OSPF является построение всеобъемлющей карты сети, одинаковой на всех маршрутизаторах. Каждый маршрутизатор периодически и при изменениях в топологии рассылает специализированные объявления о состоянии каналов (LSA). На их основе строится единое представление о структуре сети, что обеспечивает маршрутизаторам информацию обо всех доступных узлах и связях между ними. После формирования топологии для каждого узла запускается алгоритм SPF (Shortest Path First), основанный на методе Дейкстры. Этот алгоритм вычисляет кратчайшие пути до всех доступных узлов, учитывая метрики каналов и их состояние. В результате каждый маршрутизатор получает собственную таблицу маршрутизации, которая обеспечивает оптимальную доставку пакетов.

3 Результаты

3.1 Линейная топология

Сначала рассмотрим работу протокола на сети с линейной топологией. Построим граф сети, указав радиус соединения равным $r = 1.5$.

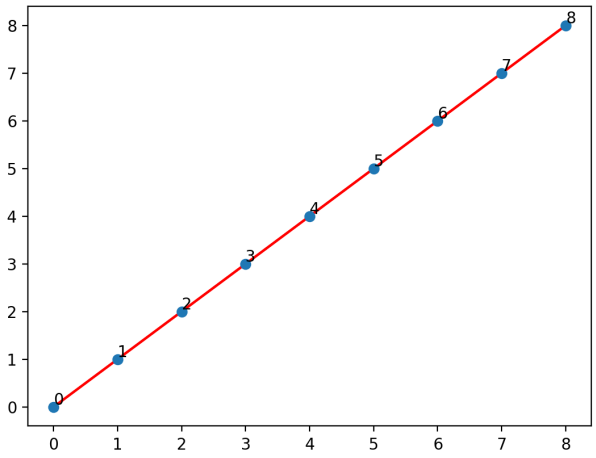

Рис. 1: Граф сети с линейной топологией

Найдём кратчайшие пути между всеми парами узлов сети.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: [0, 1]
path 0 -> 2: [0, 1, 2]
path 0 -> 3: [0, 1, 2, 3]
path 0 -> 4: [0, 1, 2, 3, 4]
path 0 -> 5: [0, 1, 2, 3, 4, 5]
path 0 -> 6: [0, 1, 2, 3, 4, 5, 6]
path 0 -> 7: [0, 1, 2, 3, 4, 5, 6, 7]
path 0 -> 8: [0, 1, 2, 3, 4, 5, 6, 7, 8]
--------------------------------------------------
Start node 1:
path 1 -> 0: [1, 0]
path 1 -> 1: [1]
path 1 -> 2: [1, 2]
path 1 -> 3: [1, 2, 3]
path 1 -> 4: [1, 2, 3, 4]
path 1 -> 5: [1, 2, 3, 4, 5]
path 1 -> 6: [1, 2, 3, 4, 5, 6]
path 1 -> 7: [1, 2, 3, 4, 5, 6, 7]
path 1 -> 8: [1, 2, 3, 4, 5, 6, 7, 8]
--------------------------------------------------
Start node 2:
path 2 -> 0: [2, 1, 0]
```

```
path 2 -> 1: [2, 1]
path 2 -> 2: [2]
path 2 -> 3: [2, 3]
path 2 -> 4: [2, 3, 4]
path 2 -> 5: [2, 3, 4, 5]
path 2 -> 6: [2, 3, 4, 5, 6]
path 2 -> 7: [2, 3, 4, 5, 6, 7]
path 2 -> 8: [2, 3, 4, 5, 6, 7, 8]
--------------------------------------------------
Start node 3:
path 3 -> 0: [3, 2, 1, 0]
path 3 -> 1: [3, 2, 1]
path 3 -> 2: [3, 2]
path 3 -> 3: [3]
path 3 -> 4: [3, 4]
path 3 -> 5: [3, 4, 5]
path 3 -> 6: [3, 4, 5, 6]
path 3 -> 7: [3, 4, 5, 6, 7]
path 3 -> 8: [3, 4, 5, 6, 7, 8]
--------------------------------------------------
Start node 4:
path 4 -> 0: [4, 3, 2, 1, 0]
path 4 -> 1: [4, 3, 2, 1]
path 4 -> 2: [4, 3, 2]
path 4 -> 3: [4, 3]
path 4 -> 4: [4]
path 4 -> 5: [4, 5]
path 4 -> 6: [4, 5, 6]
path 4 -> 7: [4, 5, 6, 7]
path 4 -> 8: [4, 5, 6, 7, 8]
--------------------------------------------------
Start node 5:
path 5 -> 0: [5, 4, 3, 2, 1, 0]
path 5 -> 1: [5, 4, 3, 2, 1]
path 5 -> 2: [5, 4, 3, 2]
path 5 -> 3: [5, 4, 3]
path 5 -> 4: [5, 4]
path 5 -> 5: [5]
path 5 -> 6: [5, 6]
path 5 -> 7: [5, 6, 7]
path 5 -> 8: [5, 6, 7, 8]
--------------------------------------------------
Start node 6:
```

```
path 6 -> 0: [6, 5, 4, 3, 2, 1, 0]
path 6 -> 1: [6, 5, 4, 3, 2, 1]
path 6 -> 2: [6, 5, 4, 3, 2]
path 6 -> 3: [6, 5, 4, 3]
path 6 -> 4: [6, 5, 4]
path 6 -> 5: [6, 5]
path 6 -> 6: [6]
path 6 -> 7: [6, 7]
path 6 -> 8: [6, 7, 8]
--------------------------------------------------
Start node 7:
path 7 -> 0: [7, 6, 5, 4, 3, 2, 1, 0]
path 7 -> 1: [7, 6, 5, 4, 3, 2, 1]
path 7 -> 2: [7, 6, 5, 4, 3, 2]
path 7 -> 3: [7, 6, 5, 4, 3]
path 7 -> 4: [7, 6, 5, 4]
path 7 -> 5: [7, 6, 5]
path 7 -> 6: [7, 6]
path 7 -> 7: [7]
path 7 -> 8: [7, 8]
--------------------------------------------------
Start node 8:
path 8 -> 0: [8, 7, 6, 5, 4, 3, 2, 1, 0]
path 8 -> 1: [8, 7, 6, 5, 4, 3, 2, 1]
path 8 -> 2: [8, 7, 6, 5, 4, 3, 2]
path 8 -> 3: [8, 7, 6, 5, 4, 3]
path 8 -> 4: [8, 7, 6, 5, 4]
path 8 -> 5: [8, 7, 6, 5]
path 8 -> 6: [8, 7, 6]
path 8 -> 7: [8, 7]
path 8 -> 8: [8]
--------------------------------------------------
```

Теперь уберём из сети узел 2 и перестроим граф сети.

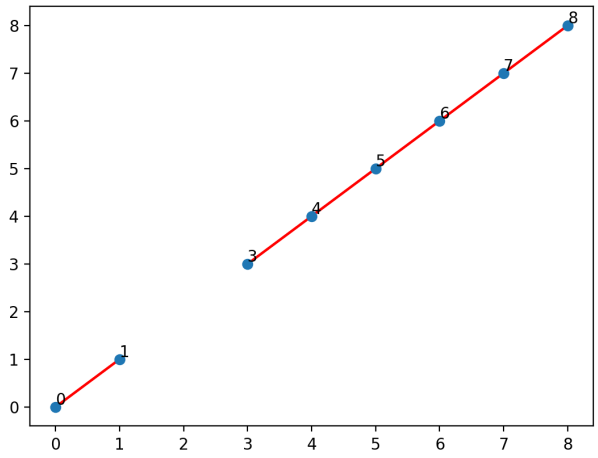

Рис. 2: Граф сети с линейной топологией без узла 2

Приведём кратчайшие пути для тех же пар узлов.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: [0, 1]
path 0 -> 2: []
path 0 -> 3: []
path 0 -> 4: []
path 0 -> 5: []
path 0 -> 6: []
path 0 -> 7: []
path 0 -> 8: []
-----------------------------------------------------
Start node 1:
path 1 -> 0: [1, 0]
path 1 -> 1: [1]
path 1 -> 2: []
path 1 -> 3: []
path 1 -> 4: []
path 1 -> 5: []
path 1 -> 6: []
path 1 -> 7: []
path 1 -> 8: []
-----------------------------------------------------
Start node 2:
path 2 -> 0: []
```

```
path 2 -> 1: []
path 2 -> 2: [2]
path 2 -> 3: []
path 2 -> 4: []
path 2 -> 5: []
path 2 -> 6: []
path 2 -> 7: []
path 2 -> 8: []
--------------------------------------------------
Start node 3:
path 3 -> 0: []
path 3 -> 1: []
path 3 -> 2: []
path 3 -> 3: [3]
path 3 -> 4: [3, 4]
path 3 -> 5: [3, 4, 5]
path 3 -> 6: [3, 4, 5, 6]
path 3 -> 7: [3, 4, 5, 6, 7]
path 3 -> 8: [3, 4, 5, 6, 7, 8]
--------------------------------------------------
Start node 4:
path 4 -> 0: []
path 4 -> 1: []
path 4 -> 2: []
path 4 -> 3: [4, 3]
path 4 -> 4: [4]
path 4 -> 5: [4, 5]
path 4 -> 6: [4, 5, 6]
path 4 -> 7: [4, 5, 6, 7]
path 4 -> 8: [4, 5, 6, 7, 8]
--------------------------------------------------
Start node 5:
path 5 -> 0: []
path 5 -> 1: []
path 5 -> 2: []
path 5 -> 3: [5, 4, 3]
path 5 -> 4: [5, 4]
path 5 -> 5: [5]
path 5 -> 6: [5, 6]
path 5 -> 7: [5, 6, 7]
path 5 -> 8: [5, 6, 7, 8]
--------------------------------------------------
Start node 6:
```

```
path 6 -> 0: []
path 6 -> 1: []
path 6 -> 2: []
path 6 -> 3: [6, 5, 4, 3]
path 6 -> 4: [6, 5, 4]
path 6 -> 5: [6, 5]
path 6 -> 6: [6]
path 6 -> 7: [6, 7]
path 6 -> 8: [6, 7, 8]
--------------------------------------------------
Start node 7:
path 7 -> 0: []
path 7 -> 1: []
path 7 -> 2: []
path 7 -> 3: [7, 6, 5, 4, 3]
path 7 -> 4: [7, 6, 5, 4]
path 7 -> 5: [7, 6, 5]
path 7 -> 6: [7, 6]
path 7 -> 7: [7]
path 7 -> 8: [7, 8]
--------------------------------------------------
Start node 8:
path 8 -> 0: []
path 8 -> 1: []
path 8 -> 2: []
path 8 -> 3: [8, 7, 6, 5, 4, 3]
path 8 -> 4: [8, 7, 6, 5, 4]
path 8 -> 5: [8, 7, 6, 5]
path 8 -> 6: [8, 7, 6]
path 8 -> 7: [8, 7]
path 8 -> 8: [8]
--------------------------------------------------
```

3.2 Кольцевидная топология

Проведём аналогичную процедуру для сети с кольцевидной топологией. Построим граф сети с радиусом соединения $r = 1.7$.

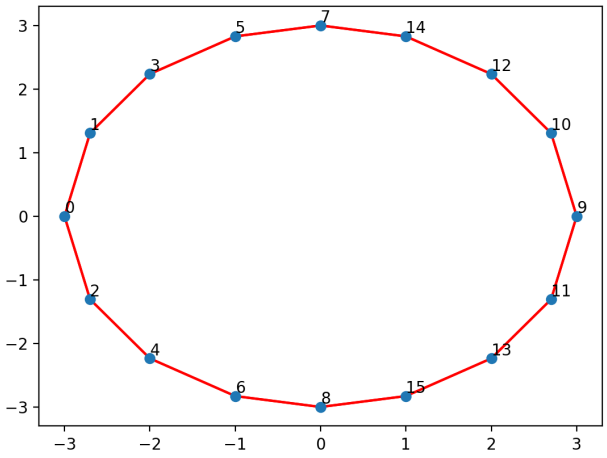

Рис. 3: Граф сети с кольцевидной топологией

Найдём кратчайшие пути между всеми парами узлов сети. Приведём некоторые примеры.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: [0, 1]
path 0 -> 2: [0, 2]
path 0 -> 3: [0, 1, 3]
path 0 -> 4: [0, 2, 4]
path 0 -> 5: [0, 1, 3, 5]
path 0 -> 6: [0, 2, 4, 6]
path 0 -> 7: [0, 1, 3, 5, 7]
path 0 -> 8: [0, 2, 4, 6, 8]
path 0 -> 9: [0, 1, 3, 5, 7, 14, 12, 10, 9]
path 0 -> 10: [0, 1, 3, 5, 7, 14, 12, 10]
path 0 -> 11: [0, 2, 4, 6, 8, 15, 13, 11]
path 0 -> 12: [0, 1, 3, 5, 7, 14, 12]
path 0 -> 13: [0, 2, 4, 6, 8, 15, 13]
path 0 -> 14: [0, 1, 3, 5, 7, 14]
path 0 -> 15: [0, 2, 4, 6, 8, 15]

--------------------------------------------------
Start node 1:
path 1 -> 0: [1, 0]
path 1 -> 1: [1]
path 1 -> 2: [1, 0, 2]
path 1 -> 3: [1, 3]
```

```
path 1 -> 4: [1, 0, 2, 4]
path 1 -> 5: [1, 3, 5]
path 1 -> 6: [1, 0, 2, 4, 6]
path 1 -> 7: [1, 3, 5, 7]
path 1 -> 8: [1, 0, 2, 4, 6, 8]
path 1 -> 9: [1, 3, 5, 7, 14, 12, 10, 9]
path 1 -> 10: [1, 3, 5, 7, 14, 12, 10]
path 1 -> 11: [1, 3, 5, 7, 14, 12, 10, 9, 11]
path 1 -> 12: [1, 3, 5, 7, 14, 12]
path 1 -> 13: [1, 0, 2, 4, 6, 8, 15, 13]
path 1 -> 14: [1, 3, 5, 7, 14]
path 1 -> 15: [1, 0, 2, 4, 6, 8, 15]
--------------------------------------------------
Start node 2:
path 2 -> 0: [2, 0]
path 2 -> 1: [2, 0, 1]
path 2 -> 2: [2]
path 2 -> 3: [2, 0, 1, 3]
path 2 -> 4: [2, 4]
path 2 -> 5: [2, 0, 1, 3, 5]
path 2 -> 6: [2, 4, 6]
path 2 -> 7: [2, 0, 1, 3, 5, 7]
path 2 -> 8: [2, 4, 6, 8]
path 2 -> 9: [2, 4, 6, 8, 15, 13, 11, 9]
path 2 -> 10: [2, 4, 6, 8, 15, 13, 11, 9, 10]
path 2 -> 11: [2, 4, 6, 8, 15, 13, 11]
path 2 -> 12: [2, 0, 1, 3, 5, 7, 14, 12]
path 2 -> 13: [2, 4, 6, 8, 15, 13]
path 2 -> 14: [2, 0, 1, 3, 5, 7, 14]
path 2 -> 15: [2, 4, 6, 8, 15]
--------------------------------------------------
Start node 3:
path 3 -> 0: [3, 1, 0]
path 3 -> 1: [3, 1]
path 3 -> 2: [3, 1, 0, 2]
path 3 -> 3: [3]
path 3 -> 4: [3, 1, 0, 2, 4]
path 3 -> 5: [3, 5]
path 3 -> 6: [3, 1, 0, 2, 4, 6]
path 3 -> 7: [3, 5, 7]
path 3 -> 8: [3, 1, 0, 2, 4, 6, 8]
path 3 -> 9: [3, 5, 7, 14, 12, 10, 9]
path 3 -> 10: [3, 5, 7, 14, 12, 10]
```

```
path 3 -> 11: [3, 5, 7, 14, 12, 10, 9, 11]
path 3 -> 12: [3, 5, 7, 14, 12]
path 3 -> 13: [3, 5, 7, 14, 12, 10, 9, 11, 13]
path 3 -> 14: [3, 5, 7, 14]
path 3 -> 15: [3, 1, 0, 2, 4, 6, 8, 15]
--------------------------------------------------
Start node 4:
path 4 -> 0: [4, 2, 0]
path 4 -> 1: [4, 2, 0, 1]
path 4 -> 2: [4, 2]
path 4 -> 3: [4, 2, 0, 1, 3]
path 4 -> 4: [4]
path 4 -> 5: [4, 2, 0, 1, 3, 5]
path 4 -> 6: [4, 6]
path 4 -> 7: [4, 2, 0, 1, 3, 5, 7]
path 4 -> 8: [4, 6, 8]
path 4 -> 9: [4, 6, 8, 15, 13, 11, 9]
path 4 -> 10: [4, 6, 8, 15, 13, 11, 9, 10]
path 4 -> 11: [4, 6, 8, 15, 13, 11]
path 4 -> 12: [4, 6, 8, 15, 13, 11, 9, 10, 12]
path 4 -> 13: [4, 6, 8, 15, 13]
path 4 -> 14: [4, 2, 0, 1, 3, 5, 7, 14]
path 4 -> 15: [4, 6, 8, 15]
--------------------------------------------------
Start node 5:
path 5 -> 0: [5, 3, 1, 0]
path 5 -> 1: [5, 3, 1]
path 5 -> 2: [5, 3, 1, 0, 2]
path 5 -> 3: [5, 3]
path 5 -> 4: [5, 3, 1, 0, 2, 4]
path 5 -> 5: [5]
path 5 -> 6: [5, 3, 1, 0, 2, 4, 6]
path 5 -> 7: [5, 7]
path 5 -> 8: [5, 3, 1, 0, 2, 4, 6, 8]
path 5 -> 9: [5, 7, 14, 12, 10, 9]
path 5 -> 10: [5, 7, 14, 12, 10]
path 5 -> 11: [5, 7, 14, 12, 10, 9, 11]
path 5 -> 12: [5, 7, 14, 12]
path 5 -> 13: [5, 7, 14, 12, 10, 9, 11, 13]
path 5 -> 14: [5, 7, 14]
path 5 -> 15: [5, 7, 14, 12, 10, 9, 11, 13, 15]
--------------------------------------------------
Start node 6:
```

```
path 6 -> 0: [6, 4, 2, 0]
path 6 -> 1: [6, 4, 2, 0, 1]
path 6 -> 2: [6, 4, 2]
path 6 -> 3: [6, 4, 2, 0, 1, 3]
path 6 -> 4: [6, 4]
path 6 -> 5: [6, 4, 2, 0, 1, 3, 5]
path 6 -> 6: [6]
path 6 -> 7: [6, 4, 2, 0, 1, 3, 5, 7]
path 6 -> 8: [6, 8]
path 6 -> 9: [6, 8, 15, 13, 11, 9]
path 6 -> 10: [6, 8, 15, 13, 11, 9, 10]
path 6 -> 11: [6, 8, 15, 13, 11]
path 6 -> 12: [6, 8, 15, 13, 11, 9, 10, 12]
path 6 -> 13: [6, 8, 15, 13]
path 6 -> 14: [6, 8, 15, 13, 11, 9, 10, 12, 14]
path 6 -> 15: [6, 8, 15]
--------------------------------------------------
Start node 7:
path 7 -> 0: [7, 5, 3, 1, 0]
path 7 -> 1: [7, 5, 3, 1]
path 7 -> 2: [7, 5, 3, 1, 0, 2]
path 7 -> 3: [7, 5, 3]
path 7 -> 4: [7, 5, 3, 1, 0, 2, 4]
path 7 -> 5: [7, 5]
path 7 -> 6: [7, 5, 3, 1, 0, 2, 4, 6]
path 7 -> 7: [7]
path 7 -> 8: [7, 5, 3, 1, 0, 2, 4, 6, 8]
path 7 -> 9: [7, 14, 12, 10, 9]
path 7 -> 10: [7, 14, 12, 10]
path 7 -> 11: [7, 14, 12, 10, 9, 11]
path 7 -> 12: [7, 14, 12]
path 7 -> 13: [7, 14, 12, 10, 9, 11, 13]
path 7 -> 14: [7, 14]
path 7 -> 15: [7, 14, 12, 10, 9, 11, 13, 15]
--------------------------------------------------
Start node 8:
path 8 -> 0: [8, 6, 4, 2, 0]
path 8 -> 1: [8, 6, 4, 2, 0, 1]
path 8 -> 2: [8, 6, 4, 2]
path 8 -> 3: [8, 6, 4, 2, 0, 1, 3]
path 8 -> 4: [8, 6, 4]
path 8 -> 5: [8, 6, 4, 2, 0, 1, 3, 5]
path 8 -> 6: [8, 6]
```

```
path 8 -> 7: [8, 6, 4, 2, 0, 1, 3, 5, 7]
path 8 -> 8: [8]
path 8 -> 9: [8, 15, 13, 11, 9]
path 8 -> 10: [8, 15, 13, 11, 9, 10]
path 8 -> 11: [8, 15, 13, 11]
path 8 -> 12: [8, 15, 13, 11, 9, 10, 12]
path 8 -> 13: [8, 15, 13]
path 8 -> 14: [8, 15, 13, 11, 9, 10, 12, 14]
path 8 -> 15: [8, 15]
--------------------------------------------------
```

После удаления узла 5 граф сети бедет имет следующий вид.

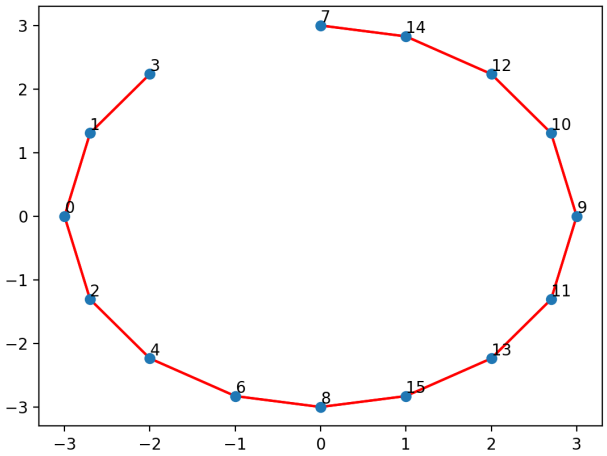

Рис. 4: Граф сети с кольцевидной топологией без узла 5

Приведём кратчайшие пути для тех же пар узлов.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: [0, 1]
path 0 -> 2: [0, 2]
path 0 -> 3: [0, 1, 3]
path 0 -> 4: [0, 2, 4]
path 0 -> 5: []
path 0 -> 6: [0, 2, 4, 6]
path 0 -> 7: [0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12, 14,
   7]
path 0 -> 8: [0, 2, 4, 6, 8]
```

```
path 0 -> 9: [0, 2, 4, 6, 8, 15, 13, 11, 9]
path 0 -> 10: [0, 2, 4, 6, 8, 15, 13, 11, 9, 10]
path 0 -> 11: [0, 2, 4, 6, 8, 15, 13, 11]
path 0 -> 12: [0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12]
path 0 -> 13: [0, 2, 4, 6, 8, 15, 13]
path 0 -> 14: [0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12, 14]
path 0 -> 15: [0, 2, 4, 6, 8, 15]
--------------------------------------------------
Start node 1:
path 1 -> 0: [1, 0]
path 1 -> 1: [1]
path 1 -> 2: [1, 0, 2]
path 1 -> 3: [1, 3]
path 1 -> 4: [1, 0, 2, 4]
path 1 -> 5: []
path 1 -> 6: [1, 0, 2, 4, 6]
path 1 -> 7: [1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12,
   14, 7]
path 1 -> 8: [1, 0, 2, 4, 6, 8]
path 1 -> 9: [1, 0, 2, 4, 6, 8, 15, 13, 11, 9]
path 1 -> 10: [1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10]
path 1 -> 11: [1, 0, 2, 4, 6, 8, 15, 13, 11]
path 1 -> 12: [1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12]
path 1 -> 13: [1, 0, 2, 4, 6, 8, 15, 13]
path 1 -> 14: [1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12,
   14]
path 1 -> 15: [1, 0, 2, 4, 6, 8, 15]
--------------------------------------------------
Start node 2:
path 2 -> 0: [2, 0]
path 2 -> 1: [2, 0, 1]
path 2 -> 2: [2]
path 2 -> 3: [2, 0, 1, 3]
path 2 -> 4: [2, 4]
path 2 -> 5: []
path 2 -> 6: [2, 4, 6]
path 2 -> 7: [2, 4, 6, 8, 15, 13, 11, 9, 10, 12, 14, 7]
path 2 -> 8: [2, 4, 6, 8]
path 2 -> 9: [2, 4, 6, 8, 15, 13, 11, 9]
path 2 -> 10: [2, 4, 6, 8, 15, 13, 11, 9, 10]
path 2 -> 11: [2, 4, 6, 8, 15, 13, 11]
path 2 -> 12: [2, 4, 6, 8, 15, 13, 11, 9, 10, 12]
path 2 -> 13: [2, 4, 6, 8, 15, 13]
```

```
path 2 -> 14: [2, 4, 6, 8, 15, 13, 11, 9, 10, 12, 14]
path 2 -> 15: [2, 4, 6, 8, 15]
--------------------------------------------------
Start node 3:
path 3 -> 0: [3, 1, 0]
path 3 -> 1: [3, 1]
path 3 -> 2: [3, 1, 0, 2]
path 3 -> 3: [3]
path 3 -> 4: [3, 1, 0, 2, 4]
path 3 -> 5: []
path 3 -> 6: [3, 1, 0, 2, 4, 6]
path 3 -> 7: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10, 12,
    14, 7]
path 3 -> 8: [3, 1, 0, 2, 4, 6, 8]
path 3 -> 9: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11, 9]
path 3 -> 10: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10]
path 3 -> 11: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11]
path 3 -> 12: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10,
   12]
path 3 -> 13: [3, 1, 0, 2, 4, 6, 8, 15, 13]
path 3 -> 14: [3, 1, 0, 2, 4, 6, 8, 15, 13, 11, 9, 10,
   12, 14]
path 3 -> 15: [3, 1, 0, 2, 4, 6, 8, 15]
--------------------------------------------------
Start node 4:
path 4 -> 0: [4, 2, 0]
path 4 -> 1: [4, 2, 0, 1]
path 4 -> 2: [4, 2]
path 4 -> 3: [4, 2, 0, 1, 3]
path 4 -> 4: [4]
path 4 -> 5: []
path 4 -> 6: [4, 6]
path 4 -> 7: [4, 6, 8, 15, 13, 11, 9, 10, 12, 14, 7]
path 4 -> 8: [4, 6, 8]
path 4 -> 9: [4, 6, 8, 15, 13, 11, 9]
path 4 -> 10: [4, 6, 8, 15, 13, 11, 9, 10]
path 4 -> 11: [4, 6, 8, 15, 13, 11]
path 4 -> 12: [4, 6, 8, 15, 13, 11, 9, 10, 12]
path 4 -> 13: [4, 6, 8, 15, 13]
path 4 -> 14: [4, 6, 8, 15, 13, 11, 9, 10, 12, 14]
path 4 -> 15: [4, 6, 8, 15]
--------------------------------------------------
Start node 5:
```

```
path 5 -> 0: []
path 5 -> 1: []
path 5 -> 2: []
path 5 -> 3: []
path 5 -> 4: []
path 5 -> 5: [5]
path 5 -> 6: []
path 5 -> 7: []
path 5 -> 8: []
path 5 -> 9: []
path 5 -> 10: []
path 5 -> 11: []
path 5 -> 12: []
path 5 -> 13: []
path 5 -> 14: []
path 5 -> 15: []
--------------------------------------------------
Start node 6:
path 6 -> 0: [6, 4, 2, 0]
path 6 -> 1: [6, 4, 2, 0, 1]
path 6 -> 2: [6, 4, 2]
path 6 -> 3: [6, 4, 2, 0, 1, 3]
path 6 -> 4: [6, 4]
path 6 -> 5: []
path 6 -> 6: [6]
path 6 -> 7: [6, 8, 15, 13, 11, 9, 10, 12, 14, 7]
path 6 -> 8: [6, 8]
path 6 -> 9: [6, 8, 15, 13, 11, 9]
path 6 -> 10: [6, 8, 15, 13, 11, 9, 10]
path 6 -> 11: [6, 8, 15, 13, 11]
path 6 -> 12: [6, 8, 15, 13, 11, 9, 10, 12]
path 6 -> 13: [6, 8, 15, 13]
path 6 -> 14: [6, 8, 15, 13, 11, 9, 10, 12, 14]
path 6 -> 15: [6, 8, 15]
--------------------------------------------------
Start node 7:
path 7 -> 0: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6, 4, 2,
   0]
path 7 -> 1: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6, 4, 2,
   0, 1]
path 7 -> 2: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6, 4, 2]
path 7 -> 3: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6, 4, 2,
   0, 1, 3]
```

```
path 7 -> 4: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6, 4]
path 7 -> 5: []
path 7 -> 6: [7, 14, 12, 10, 9, 11, 13, 15, 8, 6]
path 7 -> 7: [7]
path 7 -> 8: [7, 14, 12, 10, 9, 11, 13, 15, 8]
path 7 -> 9: [7, 14, 12, 10, 9]
path 7 -> 10: [7, 14, 12, 10]
path 7 -> 11: [7, 14, 12, 10, 9, 11]
path 7 -> 12: [7, 14, 12]
path 7 -> 13: [7, 14, 12, 10, 9, 11, 13]
path 7 -> 14: [7, 14]
path 7 -> 15: [7, 14, 12, 10, 9, 11, 13, 15]
--------------------------------------------------
Start node 8:
path 8 -> 0: [8, 6, 4, 2, 0]
path 8 -> 1: [8, 6, 4, 2, 0, 1]
path 8 -> 2: [8, 6, 4, 2]
path 8 -> 3: [8, 6, 4, 2, 0, 1, 3]
path 8 -> 4: [8, 6, 4]
path 8 -> 5: []
path 8 -> 6: [8, 6]
path 8 -> 7: [8, 15, 13, 11, 9, 10, 12, 14, 7]
path 8 -> 8: [8]
path 8 -> 9: [8, 15, 13, 11, 9]
path 8 -> 10: [8, 15, 13, 11, 9, 10]
path 8 -> 11: [8, 15, 13, 11]
path 8 -> 12: [8, 15, 13, 11, 9, 10, 12]
path 8 -> 13: [8, 15, 13]
path 8 -> 14: [8, 15, 13, 11, 9, 10, 12, 14]
path 8 -> 15: [8, 15]
--------------------------------------------------
```

3.3 Звёздная топология

Построим граф сети имеющей звёздную топологию.

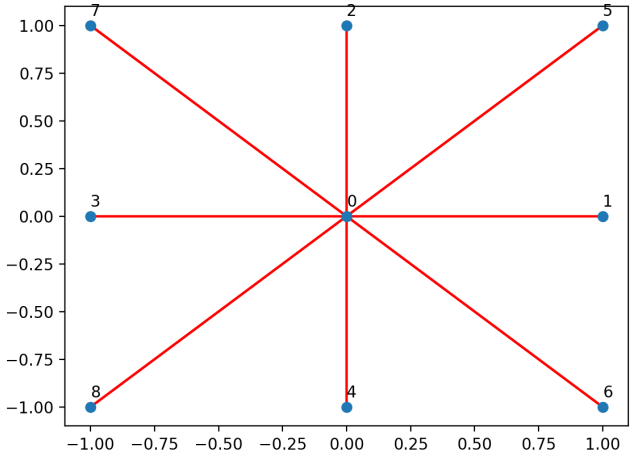

Рис. 5: Граф сети со звёздной топологией

Найдём кратчайшие пути между всеми парами узлов сети. Приведём некоторые примеры.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: [0, 1]
path 0 -> 2: [0, 2]
path 0 -> 3: [0, 3]
path 0 -> 4: [0, 4]
path 0 -> 5: [0, 5]
path 0 -> 6: [0, 6]
path 0 -> 7: [0, 7]
path 0 -> 8: [0, 8]
--------------------------------------------------
Start node 1:
path 1 -> 0: [1, 0]
path 1 -> 1: [1]
path 1 -> 2: [1, 0, 2]
path 1 -> 3: [1, 0, 3]
path 1 -> 4: [1, 0, 4]
path 1 -> 5: [1, 0, 5]
path 1 -> 6: [1, 0, 6]
path 1 -> 7: [1, 0, 7]
path 1 -> 8: [1, 0, 8]
--------------------------------------------------
Start node 2:
```

```
path 2 -> 0: [2, 0]
path 2 -> 1: [2, 0, 1]
path 2 -> 2: [2]
path 2 -> 3: [2, 0, 3]
path 2 -> 4: [2, 0, 4]
path 2 -> 5: [2, 0, 5]
path 2 -> 6: [2, 0, 6]
path 2 -> 7: [2, 0, 7]
path 2 -> 8: [2, 0, 8]
--------------------------------------------------
Start node 3:
path 3 -> 0: [3, 0]
path 3 -> 1: [3, 0, 1]
path 3 -> 2: [3, 0, 2]
path 3 -> 3: [3]
path 3 -> 4: [3, 0, 4]
path 3 -> 5: [3, 0, 5]
path 3 -> 6: [3, 0, 6]
path 3 -> 7: [3, 0, 7]
path 3 -> 8: [3, 0, 8]
--------------------------------------------------
Start node 4:
path 4 -> 0: [4, 0]
path 4 -> 1: [4, 0, 1]
path 4 -> 2: [4, 0, 2]
path 4 -> 3: [4, 0, 3]
path 4 -> 4: [4]
path 4 -> 5: [4, 0, 5]
path 4 -> 6: [4, 0, 6]
path 4 -> 7: [4, 0, 7]
path 4 -> 8: [4, 0, 8]
--------------------------------------------------
Start node 5:
path 5 -> 0: [5, 0]
path 5 -> 1: [5, 0, 1]
path 5 -> 2: [5, 0, 2]
path 5 -> 3: [5, 0, 3]
path 5 -> 4: [5, 0, 4]
path 5 -> 5: [5]
path 5 -> 6: [5, 0, 6]
path 5 -> 7: [5, 0, 7]
path 5 -> 8: [5, 0, 8]
--------------------------------------------------
```

```
Start node 6:
path 6 -> 0: [6, 0]
path 6 -> 1: [6, 0, 1]
path 6 -> 2: [6, 0, 2]
path 6 -> 3: [6, 0, 3]
path 6 -> 4: [6, 0, 4]
path 6 -> 5: [6, 0, 5]
path 6 -> 6: [6]
path 6 -> 7: [6, 0, 7]
path 6 -> 8: [6, 0, 8]
-----------------------------------------------------
Start node 7:
path 7 -> 0: [7, 0]
path 7 -> 1: [7, 0, 1]
path 7 -> 2: [7, 0, 2]
path 7 -> 3: [7, 0, 3]
path 7 -> 4: [7, 0, 4]
path 7 -> 5: [7, 0, 5]
path 7 -> 6: [7, 0, 6]
path 7 -> 7: [7]
path 7 -> 8: [7, 0, 8]
-----------------------------------------------------
Start node 8:
path 8 -> 0: [8, 0]
path 8 -> 1: [8, 0, 1]
path 8 -> 2: [8, 0, 2]
path 8 -> 3: [8, 0, 3]
path 8 -> 4: [8, 0, 4]
path 8 -> 5: [8, 0, 5]
path 8 -> 6: [8, 0, 6]
path 8 -> 7: [8, 0, 7]
path 8 -> 8: [8]
-----------------------------------------------------
```

Теперь уберём из сети центральный узел 0 и перестроим граф сети.

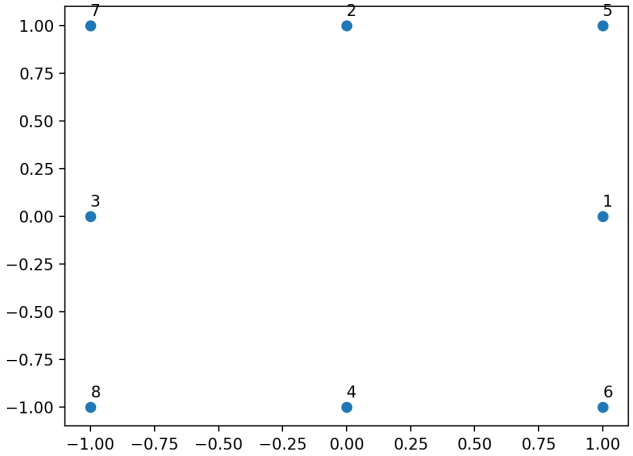

Рис. 6: Граф сети со звёздной топологией без центрального узла 0

Приведём кратчайшие пути для тех же пар узлов.

```
Start node 0:
path 0 -> 0: [0]
path 0 -> 1: []
path 0 -> 2: []
path 0 -> 3: []
path 0 -> 4: []
path 0 -> 5: []
path 0 -> 6: []
path 0 -> 7: []
path 0 -> 8: []
--------------------------------------------------
Start node 1:
path 1 -> 0: []
path 1 -> 1: [1]
path 1 -> 2: []
path 1 -> 3: []
path 1 -> 4: []
path 1 -> 5: []
path 1 -> 6: []
path 1 -> 7: []
path 1 -> 8: []
--------------------------------------------------
Start node 2:
path 2 -> 0: []
```

```
path 2 -> 1: []
path 2 -> 2: [2]
path 2 -> 3: []
path 2 -> 4: []
path 2 -> 5: []
path 2 -> 6: []
path 2 -> 7: []
path 2 -> 8: []
--------------------------------------------------
Start node 3:
path 3 -> 0: []
path 3 -> 1: []
path 3 -> 2: []
path 3 -> 3: [3]
path 3 -> 4: []
path 3 -> 5: []
path 3 -> 6: []
path 3 -> 7: []
path 3 -> 8: []
--------------------------------------------------
Start node 4:
path 4 -> 0: []
path 4 -> 1: []
path 4 -> 2: []
path 4 -> 3: []
path 4 -> 4: [4]
path 4 -> 5: []
path 4 -> 6: []
path 4 -> 7: []
path 4 -> 8: []
--------------------------------------------------
Start node 5:
path 5 -> 0: []
path 5 -> 1: []
path 5 -> 2: []
path 5 -> 3: []
path 5 -> 4: []
path 5 -> 5: [5]
path 5 -> 6: []
path 5 -> 7: []
path 5 -> 8: []
--------------------------------------------------
Start node 6:
```

```
path 6 -> 0: []
path 6 -> 1: []
path 6 -> 2: []
path 6 -> 3: []
path 6 -> 4: []
path 6 -> 5: []
path 6 -> 6: [6]
path 6 -> 7: []
path 6 -> 8: []
--------------------------------------------------
Start node 7:
path 7 -> 0: []
path 7 -> 1: []
path 7 -> 2: []
path 7 -> 3: []
path 7 -> 4: []
path 7 -> 5: []
path 7 -> 6: []
path 7 -> 7: [7]
path 7 -> 8: []
--------------------------------------------------
Start node 8:
path 8 -> 0: []
path 8 -> 1: []
path 8 -> 2: []
path 8 -> 3: []
path 8 -> 4: []
path 8 -> 5: []
path 8 -> 6: []
path 8 -> 7: []
path 8 -> 8: [8]
--------------------------------------------------
```

4 Обсуждение

В ходе лабораторной работы была реализована и протестирована работа протокола маршрутизации OSPF на различных топологиях сети: линейной, кольцевидной и звёздной.

Из полученных результатов видно, что работоспособность протокола OSPF зависит от структуры топологии сети. В линейной топологии он хорошо справляется с маршрутизацией при условии, что все узлы и кана-

лы работают стабильно, однако потеря одного узла может разделить сеть на изолированные сегменты, что резко снижает доступность ресурсов. В кольцевой топологии OSPF демонстрирует более высокую надёжность, поскольку при отказе одного узла остаётся альтернативный путь, что значительно повышает устойчивость к отказам. В звёздной топологии протокол уверенно и эффективно управляет трафиком при нормальной работе центрального узла, но потеря данного узла превращает всю сеть в несвязный набор отдельных сегментов.

5 Приложения

Репозиторий с кодом программы и кодом отчета: https://github.com/Nochoooo/networks