Санкт-Петербургский политехнический университет Петра Великого
Высшая школа прикладной математики и вычислительной физики
Кафедра прикладной математики

Лабораторная работа

по дисциплине «Компьютерные сети»

на тему

«Реализация протокола Open Shortest Path First»

Выполнил студент гр. 5040102/00201
Грицаенко Н.Д.

Преподаватель
к.ф.-м.н., доцент
Баженов А.Н.

Санкт-Петербург
2021 год

Оглавление

Постановка задачи

В данной лабораторной работе необходимо сфокусироваться на сетевом уровне по модели OSI.

Требуется смоделировать сеть из маршрутизаторов, которые обеспечивают передачу сообщений друг другу по кратчайшему пути, то есть используют протокол OSPF (*Open Shortest Path First)* [1]

Необходимо рассмотреть:

- Три вида топологии сети: линейная, кольцо, звезда.
- Перестройку таблиц достижимости при стохастических разрывах связи.

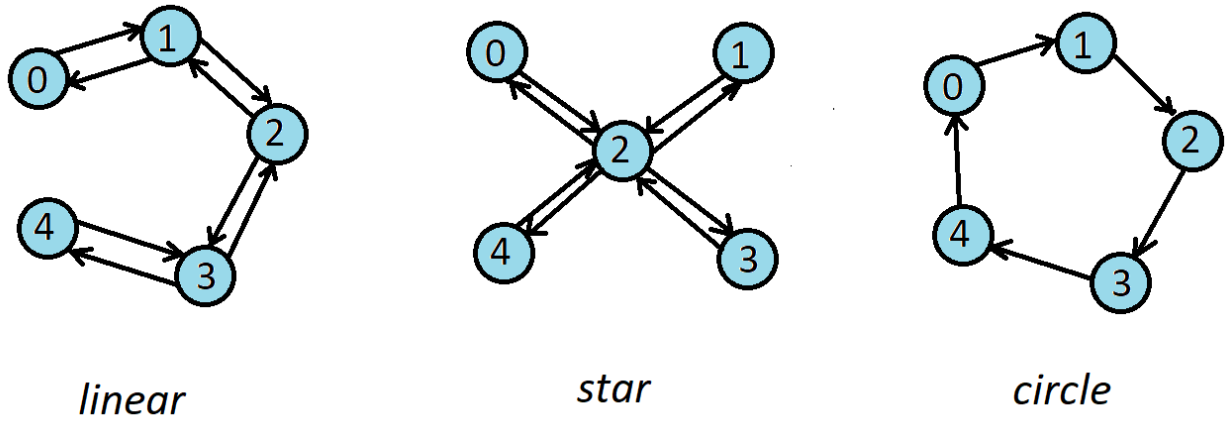

Рисунок 1. Три вида топологии сети: линейная, звезда, кольцо

Протокол динамической маршрутизации OSPF

OSPF — протокол динамической маршрутизации, основанный на технологии отслеживания состояния канала (link-state technology) и использующий для нахождения кратчайшего пути алгоритм Дейкстры [2].

Принцип работы [3] заключается в следующем:

1. После включения маршрутизаторов протокол ищет непосредственно подключенных соседей и устанавливает с ними «дружеские» отношения.
2. Затем они обмениваются друг с другом информацией о подключенных и доступных им сетях. То есть они строят карту сети (топологию). Данная карта одинакова на всех маршрутизаторах.
3. На основе полученной информации запускается алгоритм SPF (Shortest Path First, «выбор наилучшего пути»), который рассчитывает оптимальный маршрут к каждой сети.

Объявление о состоянии канала (link-state advertisement, LSA) — объявление описывает все каналы маршрутизатора, все интерфейсы и состояние каналов

Выделенный маршрутизатор (designated router, DR) — управляет процессом рассылки LSA в сети. Каждый маршрутизатор сети устанавливает отношения смежности с DR.

Информация об изменениях в сети отправляется маршрутизатором, обнаружившим это изменение, на выделенный маршрутизатор, а тот, в свою очередь, отвечает за то, чтобы эта информация была отправлена остальным маршрутизаторам.

Реализация

Для реализации был выбран язык python, среда разработки PyCharm.

Роутеры связаны с помощью орграфа с единичными весами. Также отдельно выделен designated router, который обеспечивает маршрутизацию сообщений об изменениях в топологии.

С помощью класса Thread из модуля threading все роутеры (в том числе и DR) запускаются в отдельных потоках.

Роутеры обмениваются сообщениями. Рассмотрим возможные типы сообщений.

```
class MessageType(enum.Enum):
    SEND_NEIGHBOURS = enum.auto()
    GET_TOPOLOGY = enum.auto()
    UPDATE_TOPOLOGY = enum.auto()
    REMOVE_NODE = enum.auto()
```

SEND_NEIGHBOURS – сообщение о добавлении в топологию новых соседей. Если оно приходит к DR, то он рассылает всем роутерам, кроме отправителя, то же сообщение SEND_NEIGHBOURS о необходимости добавления новых соседей.

GET_TOPOLOGY – сообщение для DR с запросом о получении текущей топологии сети. Он же в свою очередь отвечает сообщением UPDATE_TOPOLOGY, в котором присылает текущую топологию.

UPDATE_TOPOLOGY – сообщение для роутера об обновлении топологии.

REMOVE_NODE – сообщение об отключении роутера.

Пример работы: линейная топология

Рассмотрим пример работы для линейной топологии на примере сети с пятью узлами.

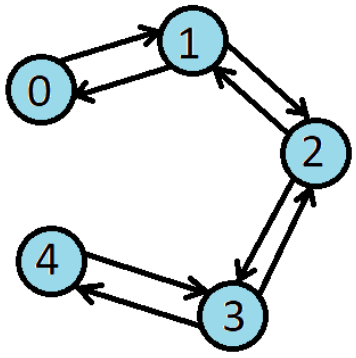

Рисунок 2. Линейная топология

$nodes$: [0, 1, 2, 3, 4],
$neighbours$: [[1], [0, 2], [1, 3], [2, 4], [3]]

Подключение роутеров к сети:

Designated router received message from router #0: (SEND_NEIGHBOURS: [1])
Designated router received message from router #0: (GET_TOPOLOGY)
Designated router received message from router #1: (SEND_NEIGHBOURS: [0, 2])
Designated router received message from router #1: (GET_TOPOLOGY)
router #1 received : (UPDATE_TOPOLOGY)
router #0 received : (UPDATE_TOPOLOGY)
router #0 received : (SEND_NEIGHBOURS: {'index': 1, 'neighbours': [0, 2]})
Designated router received message from router #2: (SEND_NEIGHBOURS: [1, 3])
Designated router received message from router #3: (SEND_NEIGHBOURS: [2, 4])
Designated router received message from router #2: (GET_TOPOLOGY)
Designated router received message from router #3: (GET_TOPOLOGY)
router #2 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2, 4]})
router #2 received : (UPDATE_TOPOLOGY)
Designated router received message from router #2: (SEND_NEIGHBOURS: [3])
router #0 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [1, 3]})
router #0 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2, 4]})
router #0 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [3]})
router #1 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [1, 3]})
router #1 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2, 4]})
router #1 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [3]})
router #3 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [1, 3]})
router #3 received : (UPDATE_TOPOLOGY)
router #3 received : (SEND_NEIGHBOURS: {'index': 2, 'neighbours': [3]})
Designated router received message from router #3: (SEND_NEIGHBOURS: [2])
router #1 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2]})
router #0 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2]})
router #2 received : (SEND_NEIGHBOURS: {'index': 3, 'neighbours': [2]})
Designated router received message from router #4: (SEND_NEIGHBOURS: [3])
Designated router received message from router #4: (GET_TOPOLOGY)
router #3 received : (SEND_NEIGHBOURS: {'index': 4, 'neighbours': [3]})
router #2 received : (SEND_NEIGHBOURS: {'index': 4, 'neighbours': [3]})
router #1 received : (SEND_NEIGHBOURS: {'index': 4, 'neighbours': [3]})
router #0 received : (SEND_NEIGHBOURS: {'index': 4, 'neighbours': [3]})
router #4 received : (UPDATE_TOPOLOGY)

Кратчайшие пути:

0: [[0], [0, 1], [0, 1, 2], [0, 1, 2, 3], [0, 1, 2, 3, 4]]

1: [[1, 0], [1], [1, 2], [1, 2, 3], [1, 2, 3, 4]]

2: [[2, 1, 0], [2, 1], [2], [2, 3], [2, 3, 4]]

3: [[3, 2, 1, 0], [3, 2, 1], [3, 2], [3], [3, 4]]

4: [[4, 3, 2, 1, 0], [4, 3, 2, 1], [4, 3, 2], [4, 3], [4]]

Теперь отключается первый роутер:

Designated router received message from router #1: (REMOVE_NODE)

router #3 received： (REMOVE_NODE: 1)

router #2 received： (REMOVE_NODE: 1)

router #4 received： (REMOVE_NODE: 1)

router #0 received： (REMOVE_NODE: 1)

Таблица достижимости перестроилась корректно:

0: [[0], [], [], [], []]

1: [[], [1], [], [], []]

2: [[], [], [2], [2, 3], [2, 3, 4]]

3: [[], [], [3, 2], [3], [3, 4]]

4: [[], [], [4, 3, 2], [4, 3], [4]]

Пример работы: кольцевая топология

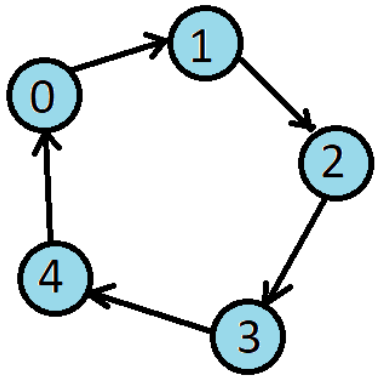

Рисунок 3. Кольцевая топология

nodes: [0, 1, 2, 3, 4],

neighbours: [[4, 1], [0, 2], [1, 3], [2, 4], [3, 0]]

Опустим в данном примере последовательность обмена сообщений, которая привела к построению кратчайших путей, поскольку оно происходит аналогично примеру с линейной топологией.

Кратчайшие пути:

0: [[0], [0, 1], [0, 1, 2], [0, 1, 2, 3], [0, 1, 2, 3, 4]]

1: [[1, 2, 3, 4, 0], [1], [1, 2], [1, 2, 3], [1, 2, 3, 4]]

2: [[2, 3, 4, 0], [2, 3, 4, 0, 1], [2], [2, 3], [2, 3, 4]]

3: [[3, 4, 0], [3, 4, 0, 1], [3, 4, 0, 1, 2], [3], [3, 4]]

4: [[4, 0], [4, 0, 1], [4, 0, 1, 2], [4, 0, 1, 2, 3], [4]]

Теперь отключается второй роутер:

Designated router received message from router #2: (REMOVE_NODE)

router #3 received： (REMOVE_NODE: 2)

router #0 received： (REMOVE_NODE: 2)

router #4 received： (REMOVE_NODE: 2)

router #1 received： (REMOVE_NODE: 2)

Таблица достижимости:

0: [[0], [0, 1], [], [], []]

1: [[], [1], [], [], []]

2: [[], [], [2], [], []]

3: [[3, 4, 0], [3, 4, 0, 1], [], [3], [3, 4]]

4: [[4, 0], [4, 0, 1], [], [], [4]]

Пример работы: топология «звезда»

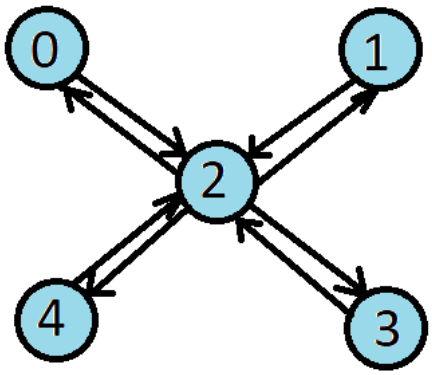

Рисунок 4. Топология "звезда"

nodes: [0, 1, 2, 3, 4]

neighbors: [[2], [2], [0, 1, 3, 4], [2], [2]]

Кратчайшие пути:

0: [[0], [0, 2, 1], [0, 2], [0, 2, 3], [0, 2, 4]]

1: [[1, 2, 0], [1], [1, 2], [1, 2, 3], [1, 2, 4]]

2: [[2, 0], [2, 1], [2], [2, 3], [2, 4]]

3: [[3, 2, 0], [3, 2, 1], [3, 2], [3], [3, 2, 4]]

4: [[4, 2, 0], [4, 2, 1], [4, 2], [4, 2, 3], [4]]

Теперь отключается центральный роутер:

Designated router received message from router #2: (REMOVE_NODE)

router #3 received： (REMOVE_NODE: 2)

router #1 received： (REMOVE_NODE: 2)

router #4 received： (REMOVE_NODE: 2)

router #0 received： (REMOVE_NODE: 2)

Таблица достижимости:

0: [[0], [], [], [], []]

1: [[], [1], [], [], []]

2: [[], [], [2], [], []]

3: [[], [], [], [3], []]

4: [[], [], [], [], [4]]

Таким образом, была реализовано моделирование протокола динамической маршрутизации OSPF со стохастическими разрывами соединения. Программа тестировалась на на трёх топологиях – «линейная», «кольцо», «звезда».

Код программы доступен на github - https://github.com/Nikitagritsaenko/Computer-Networks-labs

Использованная литература

1. А.Н. Баженов, Компьютерные сети, курс лекций
2. Нахождение кратчайших путей от заданной вершины до всех остальных вершин алгоритмом Дейкстры - http://e-maxx.ru/algo/dijkstra
3. Статья на Википедии - https://en.wikipedia.org/wiki/Open_Shortest_Path_First
4. Мануилов Г – реализация OSPF (https://github.com/dev0x13/networks_labs)