Сетевое программирование

Лекция 14

Оптимизация сетей. Оптимизация маршрутов

Усіпов Н.М., PhD · КазНУ имени аль-Фараби

Цель и план лекции

Цель

Изучение постановок задач оптимизации сетей и маршрутов, методов поиска маршрутов с ограничениями и потоков минимальной стоимости, а также приобретение навыков решения таких задач с использованием NetworkX.

1. Введение
2. Классификация задач оптимизации
3. Маршрутизация с ограничениями
4. Поток минимальной стоимости
5. Оптимизация структуры сети
6. Математическая постановка задач
7. Многокритериальная оптимизация
8. Размещение ресурсов и метаэвристики

1 Введение

- Сеть должна не только обеспечивать передачу данных, но и использовать ресурсы экономно.
- Задача оптимизации формулируется через целевую функцию и ограничения. Важно правильно определить, какие величины можно изменять: маршруты, веса каналов, объёмы потоков или саму структуру сети.

2 Классификация задач оптимизации

Оптимизация маршрутов

Топология задана, требуется выбрать пути передачи трафика.

Настройка метрик

В протоколах маршрутизации подбирают веса каналов таким образом, чтобы трафик распределялся более равномерно.

Распределение потоков

Определяется, какую часть трафика направить по каждому пути с учётом пропускной способности.

Оптимизация структуры

Определяется, какие каналы добавить или удалить, чтобы улучшить показатели при заданном бюджете.

Размещение ресурсов

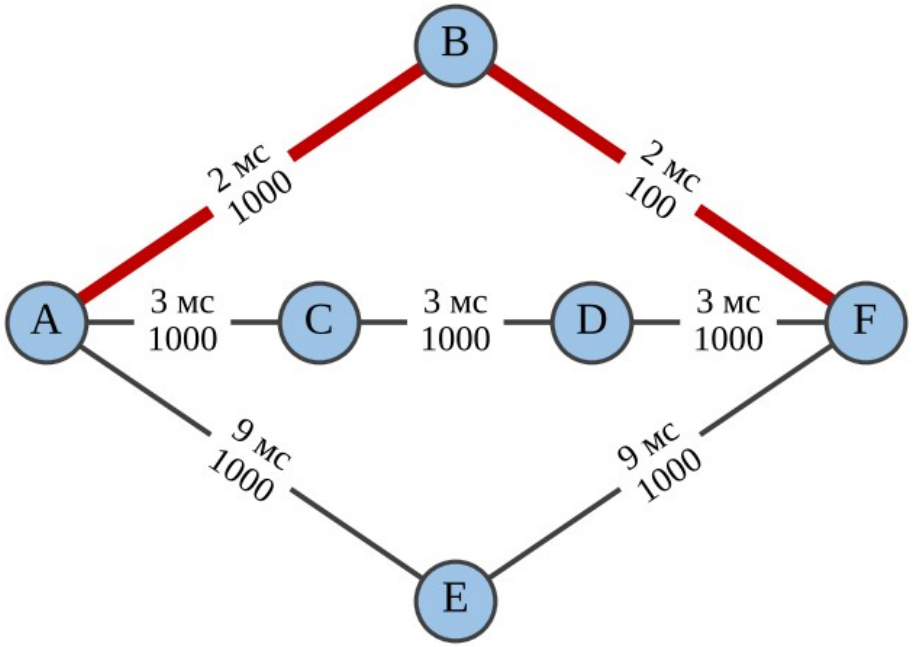
Выбираются места для установки серверов, контроллеров или кэширующих узлов.

3 Маршрутизация с ограничениями

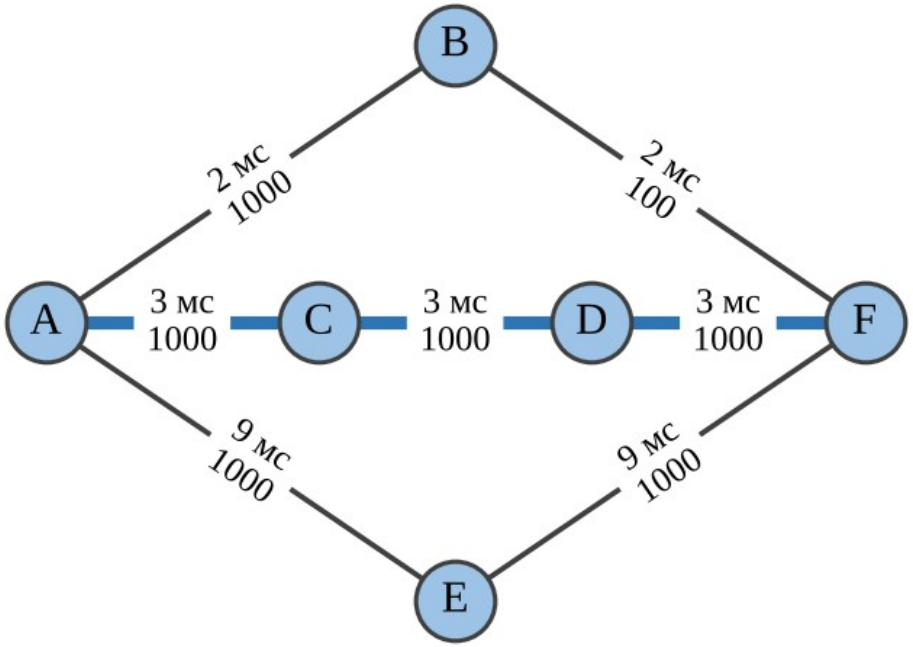
- Часто нужно найти маршрут, минимизирующий один параметр при соблюдении ограничений на другие.
- Для вогнутых (concave) параметров, таких как пропускная способность, качество маршрута определяется его самым слабым каналом.

3 Маршрутизация с ограничениями (продолжение)

Минимальная задержка



С ограничением полосы (≥ 500 Мбит/с)



Выбор маршрута без ограничений и с ограничением полосы

3 Маршрутизация с ограничениями (продолжение)

- Для аддитивных параметров, таких как задержка и стоимость, значения вдоль пути суммируются.

$$w = \alpha стоимость + \beta задержка.$$

- При этом результат зависит от выбранных коэффициентов и может не удовлетворять заданному ограничению, поэтому найденный маршрут обязательно проверяют.

3 Маршрут с ограничениями

Листинг 14.1.

```
import networkx as nx
from itertools import islice

G = nx.Graph()
for u, v, d, c in [("A", "B", 2, 1000), ("B", "F", 2, 100),
                   ("A", "C", 3, 1000), ("C", "D", 3, 1000),
                   ("D", "F", 3, 1000), ("A", "E", 9, 1000),
                   ("E", "F", 9, 1000)]:
    G.add_edge(u, v, delay=d, cap=c, cost=c / 100)

print("Без ограничений:", nx.shortest_path(G, "A", "F", "delay"))

H = G.edge_subgraph((u, v) for u, v, c in G.edges(data="cap")
                    if c >= 500)
print("Полоса >= 500:", nx.shortest_path(H, "A", "F", "delay"))

def first_within(G, s, t, weight, limit, attr, k=20):
    gen = nx.shortest_simple_paths(G, s, t, weight)
    for p in islice(gen, k):
        if nx.path_weight(G, p, attr) <= limit:
            return p
```

3 Маршрут с ограничениями (продолжение)

Листинг 14.1.

```
print("Мин. стоимость при задержке <= 10:",
      first_within(G, "A", "F", "cost", 10, "delay"))
```

3 Результат: Маршрут с ограничениями

```
Без ограничений: ['A', 'B', 'F']
Полоса >= 500: ['A', 'C', 'D', 'F']
Мин. стоимость при задержке <= 10: ['A', 'B', 'F']
```

4 Поток минимальной стоимости

- Задача о потоке минимальной стоимости обобщает задачу о максимальном потоке.

$$min \sum a(u, v)f(u, v),$$

- где a(u, v) – стоимость передачи единицы потока по дуге. В NetworkX задача решается функциями `min_cost_flow` и `network_simplex`.
- Часть трафика передаётся по дешёвому пути, а оставшаяся часть – по более дорогому.

4 Распределение трафика минимальной стоимости

Листинг 14.2.

```
D = nx.DiGraph()
D.add_node("s", demand=-20)
D.add_node("t", demand=20)
for u, v, cap, price in [("s", "a", 15, 1), ("a", "t", 15, 1),
                         ("s", "b", 20, 3), ("b", "t", 20, 2)]:
    D.add_edge(u, v, capacity=cap, weight=price)

flow = nx.min_cost_flow(D)
print("Поток:", {u: f for u, f in flow.items() if f})
print("Стоимость:", nx.cost_of_flow(D, flow))
```

```
Поток: {'s': {'a': 15, 'b': 5}, 'a': {'t': 15}, 'b': {'t': 5}}
Стоимость: 55
```

5 Оптимизация структуры сети

- При заданном бюджете на новые каналы необходимо выбирать те, которые дают наибольшее улучшение показателей.
- Также следует учитывать устойчивость решения: маршруты, оптимальные при номинальной нагрузке, могут оказаться неудовлетворительными при изменении матрицы трафика или отказе элемента.

5 Жадный выбор нового канала

Листинг 14.3.

```
R = nx.cycle_graph(12)
L0 = nx.average_shortest_path_length(R)

def gain(u, v):
    T = R.copy()
    T.add_edge(u, v)
    return L0 - nx.average_shortest_path_length(T)

best = max(nx.non_edges(R), key=lambda e: gain(*e))
print("До:", round(L0, 3))
print("Лучший канал:", best, "выигрыш:", round(gain(*best), 3))
```

```
До: 3.273
Лучший канал: (0, 6) выигрыш: 0.53
```

6 Математическая постановка задач

- Многие задачи оптимизации сетей формулируются как задачи линейного или целочисленного программирования.

$$\sum_k d_k f_e^k \le U c_e \quad \text{для всех каналов } e.$$

- Если потоки могут быть разделены произвольно, задача решается линейным программированием за полиномиальное время, например с использованием библиотеки SciPy (`linprog`) или PuLP. Если каждый поток должен следовать только по одному маршруту, переменные становятся бинарными, и задача оказывается NP-трудной.

6 Математическая постановка задач (продолжение)

- Функция стоимости иногда выбирается нелинейной. Форц и Торуп предложили кусочно-линейную выпуклую функцию, которая мало штрафует за загрузку при низких значениях и резко увеличивает штраф при загрузке, близкой к 100 %.

7 Многокритериальная оптимизация

- Часто критерии находятся в противоречии. Решение называется оптимальным по Парето, если нельзя улучшить один показатель, не ухудшив другой.
- Для задач маршрутизации с ограничением разработаны методы: LARAC, использующий релаксацию Лагранжа, и алгоритмы полностью полиномиальной аппроксимации.

8 Размещение ресурсов и метаэвристики

- Задача размещения контроллеров или кэшей может быть сформулирована как k-медианная задача: выбрать k узлов, минимизирующих суммарное расстояние пользователей до ближайшего ресурса.
- Метаэвристики применяются для задач, не имеющих эффективного точного решения.
- Применяются и методы машинного обучения, в том числе обучение с подкреплением, с помощью которого агент на основе наблюдаемого состояния сети определяет распределение трафика.

9 Разбор примера

1. Рассмотрим два маршрута: P_1 (стоимость 10, задержка 20 мс) и P_2 (стоимость 20, задержка 5 мс).
2. При равных весах w = стоимость + задержка получаем 30 и 25, поэтому выбирается P_2.
3. При w = стоимость + 0,2·задержка значения равны 14 и 21, и выбирается P_1.
4. Следовательно, выбор коэффициентов определяет результат.
5. В листинге 14.2 15 единиц идут по пути стоимостью 2, а остальные 5 единиц – по пути стоимостью 5.
6. Суммарная стоимость равна 15·2 + 5·5 = 55, что совпадает с результатом программы.

10 Вопросы для обсуждения на занятии

Цель

Что важнее для оператора: минимальная средняя задержка или минимальная максимальная загрузка?

Устойчивость

Что произойдёт с оптимальными весами после изменения матрицы трафика?

Эвристики

Как проверить качество метаэвристического решения, если неизвестен оптимум?

Контрольные вопросы

1. Какие виды задач оптимизации рассматриваются в телекоммуникационных сетях?
2. Чем ограничения вогнутого типа отличаются от аддитивных?
3. Как решается задача поиска кратчайшего пути с ограничением по полосе?
4. Почему поиск пути с двумя аддитивными критериями является сложной задачей?
5. В чём заключается задача о потоке минимальной стоимости?
6. Какое условие должны выполнять значения атрибута demand?
7. Какие методы применяются при оптимизации структуры сети?
8. Почему решение необходимо проверять при различных сценариях работы сети?

Список литературы

1. Ahuja R.K., Magnanti T.L., Orlin J.B. Network Flows: Theory, Algorithms, and Applications. – Englewood Cliffs: Prentice Hall, 1993.

2. Wang Z., Crowcroft J. Quality-of-service routing for supporting multimedia applications // IEEE Journal on Selected Areas in Communications. – 1996. – Vol. 14, No. 7. – P. 1228–1234.

3. Fortz B., Thorup M. Internet traffic engineering by optimizing OSPF weights // Proc. IEEE INFOCOM 2000. – Tel Aviv, 2000. – P. 519–528.

4. Awduche D., Malcolm J., Agogbua J., O'Dell M., McManus J. Requirements for Traffic Engineering Over MPLS: RFC 2702. – IETF, 1999.

5. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing // Science. – 1983. – Vol. 220. – P. 671–680.

6. NetworkX Documentation [Электронный ресурс]. – URL: https://networkx.org/documentation/stable/