02-27 Лекция: Алгоритмы для графов, Кратчайшие пути, MST

[Дата и время] : 2026-02-27 10:10:39

[Расположение: [Вставить расположение]] : [Вставить расположение]

[Инструктор: [Вставить имя]] : [Вставить имя]

Резюме

Лекции представляют собой обзор алгоритмов для работы с графами, охватывая три основные темы: поиск кратчайших путей, построение минимального остовного дерева (MST) и нахождение ключевых характеристик дерева (диаметр, радиус, центр).

В первой части детально разбирается алгоритм Флойда-Уоршелла для поиска путей “от всех ко всем”, его идея укорачивания пути через промежуточную вершину и способ обнаружения циклов отрицательного веса. Проводится сравнение его производительности с многократными запусками алгоритмов Беллмана-Форда и Дейкстры в зависимости от плотности графа.

Вторая часть посвящена задаче нахождения минимального остовного дерева. Вводятся определения остовного дерева и MST. Подробно описываются два ключевых алгоритма:

1. **Алгоритм Прима**, который “выращивает” дерево, на каждом шаге добавляя самое лёгкое ребро, соединяющее уже построенное дерево с новой вершиной. Обсуждается его сходство с алгоритмами Дейкстры и BFS, а также детали реализации с использованием приоритетной очереди.
2. **Алгоритм Краскала**, который сортирует все рёбра графа по весу и последовательно добавляет самые лёгкие из них, если они не образуют цикл. Для эффективного обнаружения циклов вводится и объясняется структура данных “система непересекающихся множеств” (СНМ).

Третья часть посвящена нахождению диаметра, радиуса и центра дерева с помощью последовательных запусков обхода в ширину (BFS). Сначала BFS запускается из произвольной вершины для нахождения одной из самых удалённых точек, затем от этой точки запускается второй BFS для определения диаметра, на основе которого вычисляются радиус и центральные вершины. В конце анонсируется изучение мостов и точек сочленения.

Основные положения

1. Алгоритмы поиска кратчайших путей

- **Алгоритм Флойда-Уоршелла**
  - **Назначение**: Поиск расстояний от всех вершин до всех вершин в графе.
  - **Основная идея**: Для каждой пары вершин (i, j) проверяет, нельзя ли укоротить путь между ними, пройдя через промежуточную вершину k (`d[i][k] + d[k][j] < d[i][j]`).
  - **Обнаружение отрицательных циклов**: Если после выполнения `d[i][i]` < 0 для какой-либо вершины i, это указывает на наличие цикла отрицательного веса.
  - **Сложность**: V^3.
- **Сравнение алгоритмов для поиска путей "от всех ко всем"**
  - a. **Алгоритм Флойда-Уоршелла**: Сложность V^3.
  - b. **Многократный запуск алгоритма Беллмана-Форда**: Сложность V^2E. Может быть быстрее Флойда-Уоршелла на сильно разреженных графах.
  - c. **Многократный запуск алгоритма Дейкстры**: Сложность V * (E log V). Наилучший выбор для разреженных графов без рёбер отрицательного веса.
- **Алгоритм Беллмана-Форда**
  - **Основная идея**: V-1 раз проходит по всем рёбрам графа и пытается улучшить (релаксировать) пути. На V-й итерации, если улучшение возможно, это свидетельствует о наличии отрицательного цикла.
- **Варианты реализации алгоритма Дейкстры**
  - **С приоритетной очередью**: Классическая реализация, O(E + V log V).
  - **На массиве**: Альтернативная реализация с поиском минимума в массиве, $O(V^2)$. Эффективнее на плотных графах ($E \approx V^2$).

2. Минимальное остовное дерево (MST)

- **Определение**
  - **Остовное дерево (Spanning Tree)**: Связный ациклический подграф, включающий все вершины исходного графа.

    - **Минимальное остовное дерево (MST)**: Остовное дерево с минимальной суммой весов рёбер.

- **Алгоритм Прима**

    - **Основная идея**: “Выращивает” дерево, начиная с произвольной вершины. На каждом шаге добавляет самое лёгкое ребро, соединяющее вершину в дереве с вершиной вне дерева.
    - **Сходство/Различие с Дейкстрой**: Похож на Дейкстру, но в приоритетной очереди приоритетом является вес ребра, а не суммарное расстояние от старта.
    - **Реализация**: Использует приоритетную очередь для выбора рёбер. Процесс продолжается до включения всех вершин.

- **Алгоритм Краскала**

    - **Основная идея**: Сортирует все рёбра по возрастанию веса и последовательно добавляет их в остов, если они не образуют цикл.
    - **Обнаружение циклов**: Для эффективного отслеживания компонент связности и предотвращения циклов используется **система непересекающихся множеств (СНМ)**. Ребро `(u, v)` добавляется, только если `u` и `v` принадлежат разным компонентам.
    - **Сложность**: `O(E log E)`, определяется в основном сортировкой рёбер.

3. Система непересекающихся множеств (СНМ)

- **Назначение**: Структура данных (Disjoint Set Union, DSU) для эффективного отслеживания набора непересекающихся множеств (в контексте графов — компонент связности).

- **Операции**:

    - **Поиск представителя (Find)**: Определяет, к какому множеству принадлежит элемент.
    - **Объединение (Union)**: Сливает два множества в одно.

- **Применение в алгоритме Краскала**: Изначально каждая вершина — отдельное множество. При добавлении ребра `(u, v)` проверяется, принадлежат ли `u` и `v` разным множествам. Если да, ребро добавляется, а множества объединяются.

4. Нахождение диаметра, радиуса и центра дерева

- **Алгоритм определения характеристик дерева**

    - Использует последовательность запусков обхода в ширину (BFS).

- **Шаг 1: Первый запуск BFS из произвольной вершины**
    - Цель — найти одну из самых удалённых вершин (периферическую вершину).
- **Шаг 2: Второй запуск BFS для нахождения диаметра**
    - Из найденной на первом шаге удалённой вершины запускается второй BFS. Максимальное расстояние, найденное в этом обходе, является **диаметром** дерева.
- **Шаг 3: Вычисление радиуса и нахождение центра**
    - **Радиус ®**: Вычисляется по формуле $R = \lceil D / 2 \rceil$, где D — диаметр.
    - **Центр**: Вершина (или вершины), эксцентриситет которой равен радиусу. Находится на пути диаметра на расстоянии R от любой из его конечных точек.

5. Будущие темы и анонсы

- **Поиск мостов и точек сочленения**: Будет рассмотрен на практических занятиях. Основная идея — использование времени входа и выхода из вершин при обходе графа.
- **Анонс дополнительных лекций**: Запланированы дополнительные занятия по теории графов.

Вопросы

- [Вставить вопрос/неясность]

Задания

- [ ] 1. Проанализировать и запомнить условия, при которых многократный запуск алгоритма Беллмана-Форда или Дейкстры может быть эффективнее алгоритма Флойда-Уоршелла.
- [ ] 2. Усвоить ключевое различие в логике работы алгоритмов Прима и Дейкстры.
- [ ] 3. Быть готовым словесно объяснить идею работы алгоритма Флойда-Уоршелла.
- [ ] 4. Изучить вариант реализации алгоритма Дейкстры на массиве (сложность $O(V^2)$).

- [ ] 5. Изучить и реализовать алгоритм Краскала для поиска минимального остовного дерева.
- [ ] 6. Реализовать структуру данных “система непересекающихся множеств” (СНМ) с оптимизациями.
- [ ] 7. Проанализировать разницу в подходах и производительности между алгоритмами Прима и Краскала.
- [ ] 8. Подготовиться к практическому занятию по поиску мостов и точек сочленения в графах.
- [ ] 9. Посетить будущие дополнительные лекции по теории графов.