Объектно-ориентированное программирование

Петрусевич Денис Андреевич

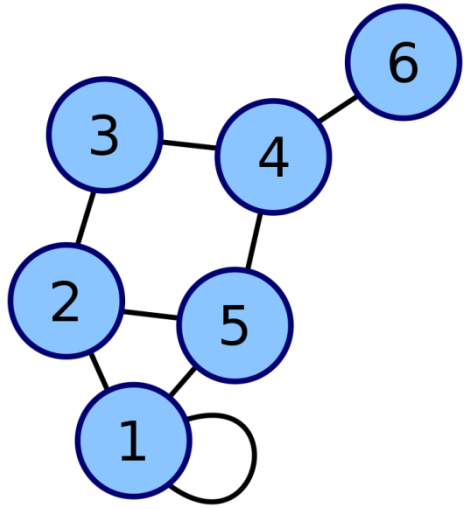
Доцент кафедры Проблем управления

petrusevich@mirea.ru

Тема 5. Представление графов

1. Список ребер
2. Матрица смежности
3. *Матрица инцидентности

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



Список ребер:

Ориентированный граф

1 1, 2, 5

2 1, 3, 5

3 2, 4…

Неориентированный граф

1 1, 2, 5

2 3, 5

3 4…

Тема 5. Представление графов

Взвешенный граф

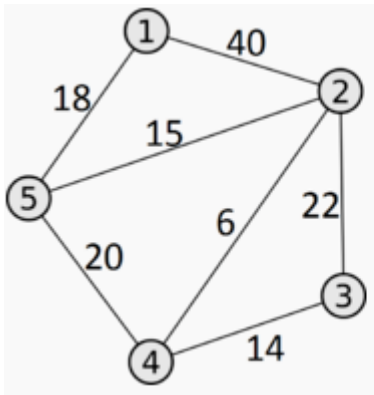
В списке ребер хранятся конечные пункты и веса

В матрице смежности вместо 0/1 храним ∞ и веса

1 (2, 40), (5, 18)

2 (3, 22), (4, 6), (5, 15)

3 (4, 14)…



$$\begin{pmatrix} 0 & 40 & \infty & \infty & 18 \\ 40 & 0 & 22 & 6 & 15 \\ \infty & 22 & 0 & 14 & \infty \\ \infty & 6 & 14 & 0 & 20 \\ 18 & 15 & \infty & 20 & 0 \end{pmatrix}$$

1 (2, 40), (5, 18)

2 (1, 40), (3, 22), (4, 6), (5, 15)

3 (2, 22), (4, 14)…

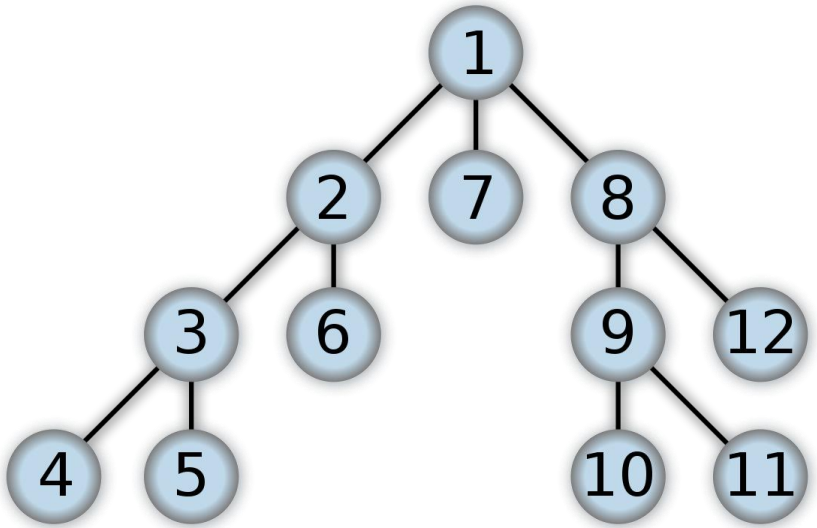
Тема 5. Поиск в глубину

Идём «вглубь» графа, по первому исследуемому пути идём до конца

Оценка по производительности: O(V + E)

V – число вершин (vertex, мн. vertices)

E – число ребер (edges)



Тема 5. Поиск в глубину

Поиск пути в невзвешенном графе

Проверка, является ли одна вершина дерева предком другой

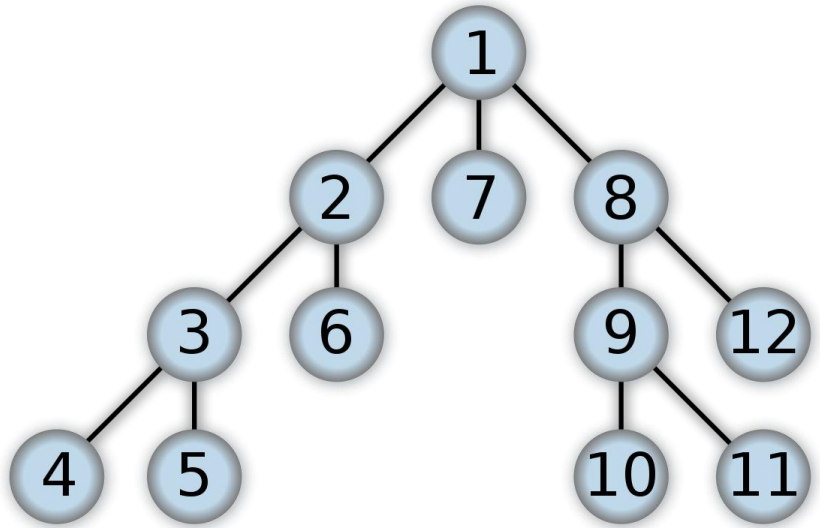
Топологическая сортировка

Проверка графа на ацикличность и нахождение цикла

Поиск компонент сильной связности

Поиск мостов

Поиск выхода из плоского лабиринта



Тема 5. Поиск в ширину

Используется для поиска кратчайшего маршрута (в ребрах) между заданной вершиной и всеми остальными

Идея похожа на распространение волны вокруг исходной точки

С небольшими модификациями превращается в алгоритм Дейкстры для взвешенного графа

Оценка по производительности: O(V + E)

Может решать те же задачи по поиску пути в невзвешенном графе, что и поиск в глубину, но можно получить полезные промежуточные результаты

Тема 5. Поиск в ширину

- Поиск **кратчайшего пути** в невзвешенном графе.
- Поиск **компонент связности** в графе
- Нахождения решения какой-либо задачи (игры) **с наименьшим числом ходов**
- Нахождение **кратчайшего цикла** в ориентированном невзвешенном графе
- Найти все рёбра / вершины, лежащие **на каком-либо кратчайшем пути** между заданной парой вершин . кратчайший путь из стартовой вершины в конечную, с чётностью, равной 0.

Тема 5. Взвешенные графы

- Поиск в ширину даст кратчайший путь только в количестве ребер, но не минимизирует суммарный вес ребер
- Применяются алгоритмы Дейкстры, Беллмана-Форда, Флойда и т.д.
- Чаще всего ищут путь между 0-й вершиной и i-й или 0-й вершиной и всеми остальными

Тема 5. Алгоритм Дейкстры

- Модификация поиска в ширину
- Можем снова заходить в уже посещенные вершины: если идем по пути, который меньше найденного ранее
- Используется процедура релаксации ребра
- Обрабатывается вершина с наименьшей текущей стоимостью пути (в отличие от реализованной на семинаре версии)

Тема 5. Алгоритм Дейкстры

Релаксация ребра

Текущий минимальный путь к вершине v: d(v)

Текущий обрабатываемый путь к смежной вершине u: d(u)

Вес ребра между вершинами u и v: w(u, v)

d(v) > d(u) + w(u,v) ? //найден лучший маршрут

d(v) = d(u) + w(u,v) //обновление

обработать вершину v

Тема 5. Алгоритм Дейкстры

Структура данных для хранения вершин	Оценка производительности
Массив	$O(V^2 + E)$
Двоичная куча	$O[(E + V)\log V]$
Фибоначчиева куча	$O(V\log V + E)$

Тема 5. Алгоритм Беллмана-Форда

Ищем путь от 0-й вершины до всех остальных

Релаксация ребра

d(v) > d(u) + w(u,v) ? //найден лучший маршрут

d(v) = d(u) + w(u,v) //обновление

обработать вершину v

Повторять это действие

Сколько раз?

Тема 5. Алгоритм Беллмана-Форда

Максимальный путь (по количеству ребер) в графе из V вершин без циклов: V - 1 ребер

Цикл положительного веса увеличивает путь

Цикл отрицательного веса обрабатывается отдельно

V = 5

Для учета максимально возможного (по числу ребер) пути (по суммарному весу он может и не быть максимальным) нужно V-1 операций релаксаций ребра

На i-й итерации промежуточный результат: минимизированы все пути, состоящие из i ребер

Тема 5. Алгоритм Беллмана-Форда

Идея для оптимизации: сортировка ребер

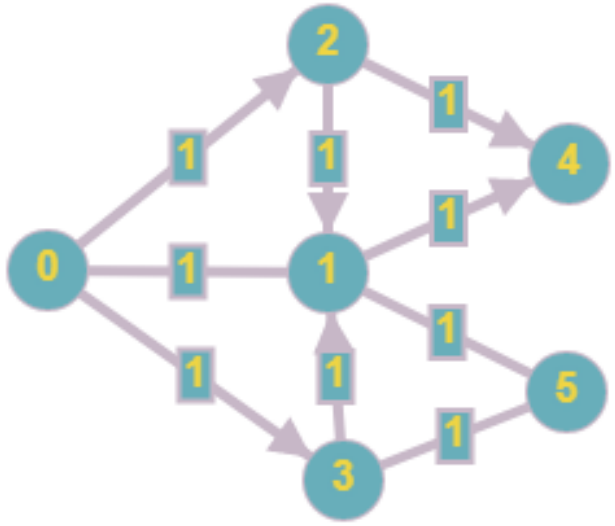
```
int operator<(Edge e1, Edge e2)
{
      if (e1.from < e2.from)
            return true;
       if (e1.from == e2.from)
             return e1.to < e2.to;
      return false;
}
```

From	To
0	1
1	2
2	3
3	4

Тема 5. Алгоритм Беллмана-Форда

Идея для оптимизации: сортировка ребер

From	To
0	1
0	2
0	3
0	1
1	0
1	4
2	1
2	4
3	1
3	5
5	1
5	3



Тема 5. Алгоритм Беллмана-Форда

Идея для оптимизации: сортировка ребер

Работу алгоритма можно заканчивать, когда перестают меняться оценки путей

Максимально нужно V – 1 итераций

Если на V-й итерации продолжают изменяться пути, в графе есть цикл отрицательного веса

Если есть отрицательный цикл, релаксацию можно проводить бесконечно. Отсюда ограничение на V – 1 итерацию

На V-1 итерации просматривается E ребер => вычислительная сложность O(VE)

Тема 5. Оценки алгоритмов

Алгоритм	Оценка	V>>E (разреженный)	E>>V (плотный)	Примечание
Поиск в глубину / ширину	O(V + E)	O(V)	O(E)	Только на невзвешенных графах / поиск минимального пути в числе ребер
Алгоритм Дейкстры	O(Vlog V + E)	O(Vlog V)	O(E)	Не обрабатывает циклы с отрицательными весами
Алгоритм Беллмана-Форда	O(VE)	O(V)	O(E)	Находит циклы с отрицательными весами
Алгоритм Флойда*	$O(V^3)$			Находит минимальные пути между всеми парами вершин

Тема 5. Минимальный остов

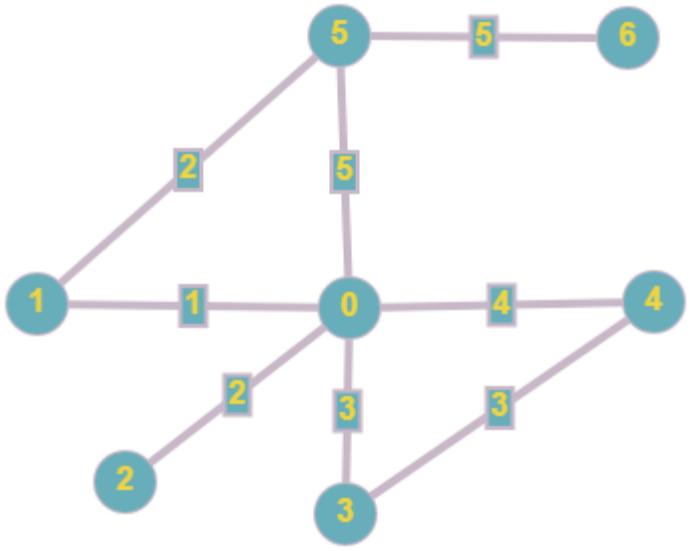
Даны вершины и возможные ребра

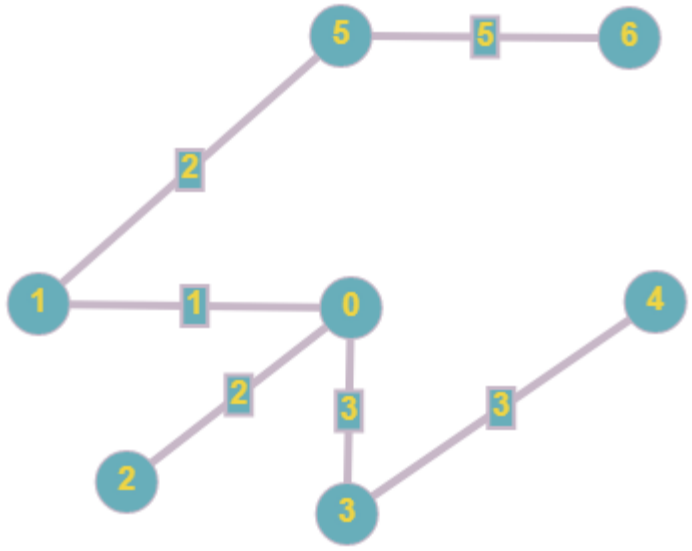
Минимальное остовное дерево: построить граф, связывающий все вершины, с минимальным суммарным весом (минимальной суммой весов ребер)

Все вершины должны оказаться связными

Цикл содержал бы лишние ребра

Минимальный остов – это дерево (связный граф без циклов)





Тема 5. Алгоритм Прима

Храним список использованных вершин

Храним список ребер, которые из них выходят

Ищем минимальное ребро и удаляем его из списка

Если ребро ведет в несвязанную вершину, добавить вершину в список / сохранить ребро в финальном списке ребер

Сравнение ребер

```
int operator< (Edge e1, Edge e2)
{return e1.weight < e2.weight;}
```

Тема 5. Алгоритм Прима

Идеи для оптимизации

В какой структуре хранить ребра, если нужно:

1) искать и удалять из неё минимум,
2) добавлять ребра?

Кучи, сбалансированные деревья?

Наивная реализация с просмотром всех исходящих ребер на каждой итерации: O(VE)

Тема 5. Алгоритм Прима

Наивная реализация: O(VE)

Плотные графы, E>>V: $O(V^2)$

Разреженные графы, V>>E: O(E log V)

Тема 5. Алгоритм Крускала

Идеи для оптимизации

Алгоритм Крускала – взгляд на задачу со стороны ребер

Сначала все вершины образуют свои компоненты связности

Отсортировать список ребер

Выбрать и удалить минимальное ребро

Проверка: связывает ли оно две компоненты? Связать компоненты, добавить ребро

Повторять, пока есть ребра / число связных компонент > 1

Оценка производительности: $O(V^2 + E \log V)$

При использовании Disjoint Set Union (система непересекающихся множеств, а множество, чаще всего, - сбалансированное дерево) : $O(E \log V)$

Литература

Поиск в глубину / ширину

Кормен. Часть VI

https://habr.com/ru/post/331192/

https://evileg.com/ru/post/512/

https://evileg.com/ru/post/494/

https://e-maxx.ru/algo/dfs

https://e-maxx.ru/algo/bfs

https://habr.com/ru/post/469967/

https://foxford.ru/wiki/informatika/algoritm-poiska-v-glubinu

Литература

Алгоритм Дейкстры

Кормен. Часть VI

https://informatics.mccme.ru/mod/statements/view3.php?id=193&chapterid=1967

https://informatics.mccme.ru/mod/statements/view3.php?id=10031&chapterid=96

https://informatics.mccme.ru/mod/statements/view3.php?id=10031&chapterid=98

https://prog-cpp.ru/deikstra/

https://e-maxx.ru/algo/dijkstra

https://www.e-olymp.com/ru/blogs/posts/21

Литература

Алгоритм Беллмана-Форда

Кормен. Часть VI

https://www.youtube.com/watch?v=hxMWBBCpR6A

https://www.geeksforgeeks.org/bellman-ford-algorithm-simple-implementation/

https://informatics.mccme.ru/mod/statements/view.php?id=10844

https://e-maxx.ru/algo/ford_bellman

https://habr.com/ru/post/201588/

Литература

Алгоритм Флойда (-Уоршелла)

Кормен. Часть VI

https://e-maxx.ru/algo/floyd_warshall_algorithm

https://www.youtube.com/watch?v=HwK67u7zaEE

https://habr.com/ru/post/105825/

https://foxford.ru/wiki/informatika/algoritm-floyda

Литература

Кормен. Часть VI

Алгоритм Прима

https://informatics.mccme.ru/mod/statements/view3.php?id=26153&chapterid=605

https://www.youtube.com/watch?v=vPHUm874EoA

https://e-maxx.ru/algo/mst_prim

https://brestprog.by/topics/mst/

http://www.hpcc.unn.ru/?dir=1051

Алгоритм Крускала

https://www.youtube.com/watch?v=mPObw3cJoTs

https://studfile.net/preview/7511620/

https://e-maxx.ru/algo/mst_kruskal

https://informatics.mccme.ru/mod/statements/view3.php?id=26153&chapterid=1377

Спасибо за внимание!

Петрусевич Денис Андреевич

Доцент Кафедры Высшей математики

petrusevich@mirea.ru