**Меняйлов Владислав Олегович**

Группа ИС-641

emindanew@yandex.ru

# Постановка задачи

Реализовать алгоритм Дейкстры поиска кратчайшего пути от заданной вершины до всех остальных. Для хранения длин кратчайших путей (массив d[i]) следует использо- вать бинарную кучу (min-heap). Для хранения графа использовать матрицу смежности.

# Описание алгоритма

Алгоритм Дейкстры(англ.*Dijkstra’s algorithm*) — алгоритм на графах, изобретённый нидерландским учёным Эдсгером Дейкстрой в 1959 году. Находит кратчайшие пути от одной из вершин графа до всех остальных. Алгоритм работает только для графов без рёбер отрицательного веса.

В начале алгоритма расстояние для начальной вершины полагается равным нулю, а все остальные расстояния заполняются большим положительным числом (большим максимального возможного пути в графе). Массив флагов заполняется нулями. Затем запускается основной цикл.

На каждом шаге цикла мы ищем вершину u с минимальным расстоянием и флагом равным нулю. Затем мы устанавливаем в ней флаг в 1 и проверяем все соседние с ней вершины *u*. Если в них (в *u*) расстояние больше, чем сумма расстояния до текущей вершины и длины ребра, то уменьшаем его. Цикл завершается, когда флаги всех вершин становятся равны 1, либо когда у всех вершин c флагом 0. *d*[*i*] = *∞* последний случай возможен тогда и только тогда, когда граф *G* несвязный.

Оценка сложности:

Tдейкстры = *O*(*n* log *n* + *m* log *n*)

d[i] хранятся в бинарной куче(Binary heap)

Tдейкстра = *O*(*m* log *n*)

**Binary heap**

*T*create = *O*(1)

*T*min = *O*(1)

*Ti*nsert =*O*(log *n*)

*T*heapify = *O*(log *n*)

*Tde*crease = *O*(log *n)*

# Организация экспериментов

* Эксперименты проводились на виртуальной машине   
  (Intel(R) Core(TM) i3-3110M CPU @2.40GHz 2.40GHz Ozu 4gb
* Операционная система Opensuze 20 x86\_64 (компилятор gcc 4.8.2)
* Ключи компиляции программы (см. README): -Wall -O2

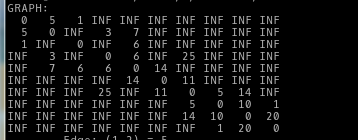
# Экспериментальное исследование

Сформировать связный граф из 10 вершин и найти в нем кратчайший путь от вершины 1 до всех остальных. Граф и найденные кратчайшие пути включить в отчет. На экран вывести путь и его длину.

**Задание 1 :** Построить граф используя матрицу смежности.

Матрица смежности графа, где INF = ∞

Матрица смежности - это матрица n\*n элементов.



Матрица 10x10 = n\*n.

Характеристики:

* Объем требуемой памяти O(|V^2|)
* Быстрое определение присутствия ребра (i,j) в графе
* За время O(1) получаем доступ к элементу аij матрицы

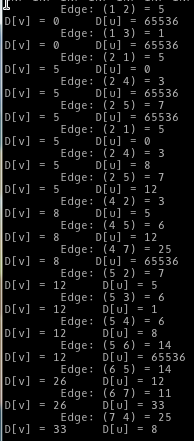
**Задание 2:** Показать пройденные пути, при определении кратчайших дорог.

При поиске кратчайших путей использовался алгоритм Дейкстры.

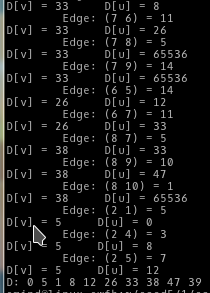
Где D[v] – Сохраненное минимальное расстояние.

D[u] – Текущие расстояние до точки

Edge ( первая вершина , вторая вершина ) = Расстояние ребра от первой вершины до второй.



**Задание 3:** Вывести кратчайшие пути до вершин.



D: = отчет начинается с 1 вершины , то есть до вершины 2 мин путь равен 5 .

# Вывод

Был реализован Алгоритм Дейкстры, также были реализованы алгоритмы для работы с бинарной кучей и графами. Были рассчитаны, с помощью алгоритма Дейкстры, минимальные пути от вершины 1 к другим вершинам.

**Ответы на вопросы:**

1.Основные шаги алгоритма Дейкстры:

1. Устанавливаем расстояние от начальной вершины до всех остальных в бесконечность .

2. Расстояние до начальной вершины = 0

3.Помещаем все вершины с приоритетом Q(min-heap): приоритет вершины i это значение D[i]

4. Запускается цикл из числа вершин

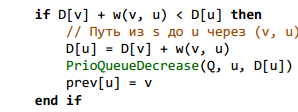
{

5. Извлекаем из очереди Q вершину v ближайшую к S

6. Отмечаем ее как посещенную

7. Проверяем стали ли пусти из s через вершину v короче

}



2. Вычислительная сложно реализованного алгоритма Дейкстры

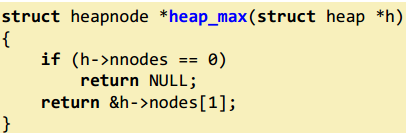
Оценка сложности:

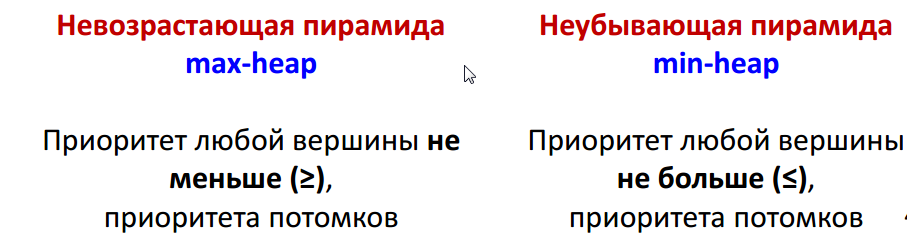
Tдейкстры = O(n log n + m log n)

d[i] хранятся в бинарной куче(Binary heap)

Tдейкстра = O(m log n)

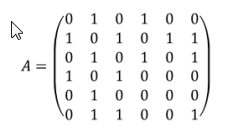
3. Операции над бинарной кучей (min-heap,max-heap)





4. Способы представления графов в памяти ( матрица смежности , списки смежных вершин )

Матрица смежности – эффективна для насыщенных графов



Списки смежных вершин – эффективен для разреженных графов

