**Лабораторная работа №** **7**

**Тема:** Алгоритмы работы с графами. Алгоритм Прима-Крускала и Дейкстры

**цель:** Научиться реализовывать алгоритм Прима-Крускала для построения каркас ого деревьев а графу и алгоритм Дейкстры для поиска кратчайшего пути

*Граф* - (В программировании) это динамическая структура данных, состоящая из совокупности объектов и связей между ними. Объекты называются узлами или вершинами, а связи между ними - ребрами или дугами.

**Способы представления графов в компьютерных программах**

Существует достаточно большое число различных способов представления графов. Рассмотрим наиболее удобные с точки зрения программирования.

**матрица смежности**

*матрица смежности* Sm - Это квадратная матрица размером NxN (N - количество вершин в графе), заполненная единицами и нулями по следующему правилу:

если в графе есть ребро e, что соединяет вершины u и v, то Sm [u, v] = 1, в противном случае Sm [u, v] = 0.

Заметим, что данное определение подходит как для ориентированных, так и для неориентированных граф и в: матрица смежности для неориентированного графабудет симметричной относительно своей главной диагонали, а для орграфа - Несимметричной.

задать взвешенный граф при помощи матрицы смежности тоже возможно. Необходимо лишь внести небольшое изменение в определение:

если в графе есть ребро e, что соединяет вершины u и v, то Sm [u, v] = v aga (e), в противном случае Sm [u, v] = 0.

Небольшое осложнения возникнет в том случае, если в графе разрешаются ребра с весом 0. Тогда надо хранить два массива: один с нулями и единицами, которые служат показателем наличия ребер, а второй - с весами этих ребер.

удобство матрицы смежности заключается в наглядности и прозрачности алгоритмов, основанных на ее использовании. А неудобство - в несколько завышенной требованию к памяти: если граф далек от полного, то в массиве, сохраняет матрицу смежности, возникает много "пустых мест" (нулей). Кроме того, для "общения" с пользователем этот способ представления графов Не слишком удобен: его лучше применять только для внутреннего представления данных.

На рисунке 1 приведены два примера матриц смежности для неориентированного и ориентированного графа.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **a** | **b** | **c** |
| **a** | 0 | 2 | 5 |
| **b** | 2 | 0 | 7 |
| **c** | 5 | 7 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | 0 | 1 | 0 | 1 | 0 | 0 |
| **2** | 0 | 1 | 0 | 0 | 1 | 0 |
| **3** | 0 | 0 | 0 | 0 | 1 | 1 |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 |
| **5** | 0 | 0 | 0 | 1 | 0 | 0 |
| **6** | 0 | 0 | 1 | 0 | 0 | 0 |

Рисунок 1 - Примеры матриц смежности взвешенного неориентированного графа (слева) и невзвешенного ориентированного графа (справа)

**список ребер**

Этот способ задания графов наиболее удобный для внешнего представления входных данных. Пусть каждая строка входного файла содержит информацию об одном ребро:

<номер\_початковои\_вершины> <номер\_кинцевои\_вершины> [<вага\_ребра>]

Примеры списков ребер изображены на рисунке 2.

|  |  |
| --- | --- |
| a, b | 2 |
| a, c | 5 |
| b, c | 7 |

|  |
| --- |
| 1, 2 |
| 1, 4 |
| 2, 2 |
| 2, 5 |
| 3, 5 |
| 3, 6 |
| 5, 4 |
| 6, 3 |

Рисунок 2 - Примеры списков ребер взвешенного неориентированного графа (слева) и невзвешенного ориентированного графа (справа)

Если задается ориентирован граф, то номера вершин понимаются как упорядоченная пара, а если граф неориентированный - как неупорядоченная.

**списки смежности**

Этот способ задания графов заключается в том, что для каждой вершины указывается список всех смежных с ней вершин (для орграфа - список вершин, является концами выходных ребер). Конкретный формат входного файла содержит списки смежности, необходимо указывать отдельно. Например, в нашем примере начальнаявершина отделена от список смежности двоеточие, а весов и ребер (во взвешенном графе) указан и в круглых скобках:

<номер\_початковои\_вершины> <номер\_сумижн й \_вершин и 1> [(<весов а ребра>)], <номер\_сумижн й \_вершин и 2> [(<весов а ребра>)] ...

Наиболее естественно применять этот способ для задания орграф и в, однако и для других вариантов он тоже подходит.

Примеры списков смежности изображены на рисунке 3.

|  |
| --- |
| a: b (2), с (5) |
| b: a (2), с (7) |
| c: a (5), b (7) |

|  |
| --- |
| 1: 2, 4 |
| 2: 2, 5 |
| 3: 5, 6 |
| 5: 4 |
| 6: 3 |

Рисунок 3 - Примеры списков смежности взвешенного неориентированного графа (слева) и невзвешенного ориентированного графа (справа)

**иерархический список**

Этот способ представления графов является всего лишь внутренней реализацией список смежности: в одном линейном списке содержатся номера начальныхвершин, а в других - номера смежных вершин или указатели на эти вершины.

**Минимальное каркасное дерево графа**

*каркасное* *дерево* связанного неориентированного графа - Ациклический связный подграф этого графа, содержащий все его вершины. Неформально говоря,каркасное дерево состоит из некоторого подмножества ребер графа, таких, что двигаясь этими ребрами можно с любой вершины графа попасть в любой другой.

*Минимальное* *каркасное* *дерево* - Это каркас графа, имеет минимальную возможную вес, где под тяжестью дерева понимается сумма весов ребер, входящих в него.

Рисунок 4 - Пример г раф а с минимальным каркасным деревом. Суммарный вес изображенного минимального каркасного дерева ре в на 37 Это минимальное каркасное дерево не уникально: удалением ребра (c, d) и добавлением ребра (a, b) получается но ве м и н и минимальными автомобиль к асне дерево

**А** **лгоритм Прима-Крускала** - а лгоритм построения минимального каркасного дерева взвешенного связного неориентированного графа.

**Пример 1.** Рассмотрим прикладную задачу, для решения которой необходимо построить минимальное каркасное дерево.

В Кировоградской области нужно соединить N сел оптоволоконными линиями связи для расширения сети Интернета. Все расстояния между селами известны.Найти наименьшую возможную длину линии, которая соединит все N сел.

Рисунок 5 - Граф узлы которого - села, а ребра - расстояния между ними

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 6 - Матрица смежности графа из рисунка 5

Раскрашиваем все в узлы графа в разный цвет. Для простоты обозначим цвета произвольными числами (рис. 7).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **В** **ершина** | 1 | 2 | 3 | 4 | 5 |
| **К** **ол** **и** **г.** | 1 | 2 | 3 | 4 | 5 |

Рисунок 7 - Матрица цветов вершин графа

В матрице смежности данного графа будем рассматривается ты только ее часть выделенную на рисунке 8.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 8 - Элементы матрицы смежности графа, с которыми будет работать алгоритм Прима-Крускала

Далее в цикле пока все вершины не будут раскрашены в один цвет повторяем следующие действия.

**1 Итерация цикла.** Находим в выделенной части ребро с минимальным весом.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 9

Это ребро (1, 4), вес которого равен 2. Сохраняем название ребра.

Обозначаем узлы 1 и 4 одинаковым цветом - они теперь часть будущего минимального каркасного дерева, которое мы строим.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вершина** | 1 | 2 | 3 | 4 | 5 |
| **цвет** | 1 | 2 | 3 | 1 | 5 |

Рисунок 10

**2** **Итерация цикла.** Находим в выделенной части ребро с минимальным весом среди ребер с разноцветными вершинами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 11

Это ребро (1, 2), вес которого равен 2. Сохраняем название ребра.

Обозначаем узлы 1 и 2 одинаковым цветом.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вершина** | 1 | 2 | 3 | 4 | 5 |
| **цвет** | 1 | 1 | 3 | 1 | 5 |

Рисунок 12

**3** **Итерация цикла.** Находим в выделенной части ребро с минимальным весом среди ребер с разноцветными вершинами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 13

Это ребро (2, 3), вес которого равен 3. Сохраняем название ребра.

Обозначаем узлы 1 и 2 одинаковым цветом.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вершина** | 1 | 2 | 3 | 4 | 5 |
| **цвет** | 1 | 1 | 1 | 1 | 5 |

Рисунок 1 апреля

**4** **Итерация цикла.** Находим в выделенной части ребро с минимальным весом среди ребер с разноцветными вершинами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 3 | 5 | 2 | 7 |
| **2** | 3 | 0 | 3 | 7 | 5 |
| **3** | 5 | 3 | 0 | 10 | 4 |
| **4** | 2 | 7 | 10 | 0 | 4 |
| **5** | 7 | 5 | 4 | 4 | 0 |

Рисунок 1 мая

Это ребро (3, 5), вес которого равен 4. Сохраняем название ребра.

Обозначаем узлы 1 и 2 одинаковым цветом.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вершина** | 1 | 2 | 3 | 4 | 5 |
| **цвет** | 1 | 1 | 1 | 1 | 1 |

Рисунок 6 января

Теперь все узлы графа раскрашены в одинаковый цвет, то есть выполнено условие выхода из цикла.

После выполнения цикла м и получили следующий список ребер: (1, 4), (1, 2), (2, 3), (3, 5), на основе него строим каркасное дерево изображенное на рисунке 17.

Рисунок 17 - минимальное каркасное дерево для графу из рисунка 2, его суммарный вес равен 12

**результат:** Оптоволоконную линию нужно провести между следующими парами с и л (1, 4), (1, 2), (2, 3), (3, 5). Необходимая длина провода равна 12 условнымединицам расстояния.

**алгоритм Дейкстры**

В теории графов, **задача о кратчайшем пути** состоит в нахождении такого пути между двумя вершинами (или узлами) графу, что сумма весов ребер, из которых он состоит, минимальна. Примером может быть нахождение кратчайшего пути между двумя пунктами на дорожной карте; в этом случае, вершинами есть пункты, а ребрами - отрезки дороги с весами, равными времени, необходимом для преодоления этого отрезка. Другой пример нахождения оптимального пути следования ИР-пакет в сети Интернет. В таком случае узлами будут маршрутизатор и, ребрами - связи между ними, а весами ребер - пропускная способность соединений.

базовые алгоритмы нахождения кратчайших путей в взвешенных графах: а лгоритм Дейкстры, а лгоритм Флойда-Уоршелла, а лгоритм Форда-Беллмана.

**Принцип работы алгоритма Дейкстры**

А лгоритм Дейкстры с находит кратчайшее расстояние от одной вершины ко всем остальным. Все веса ребер должны быть положительными числами.

Каждой вершине графа ставится в соответствие метка mark, содержащий минимальное расстояние от первой вершины к текущей вершины, минимальный путь от первой до текущей вершины хранится в массиве L. В начале все метки равны ∞. Алгоритм состоит в том, что на каждом шагу перебираются все возможные варианты продвижения по графу. При нахождении более короткого пути от вершины V 1 до вершины V 2, значение метки вершины V 2 замен й ется на длину этого пути, и так до тех пор, пока не будет пройден весь граф. В конце работы программы в массиве L будут содержаться кратчайшие пути от первой вершины до всех остальных вершин.

**Пример** **2.** Рассмотрим прикладную задачу, для решения которой необходимо найти кратчайший на расстоянии ь в графе между двумя вершинами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** | **E** |
| **A** | 0 | 78 | 30 | 50 | 0 |
| **B** | 78 | 0 | 10 | 0 | 5июня |
| **C** | 30 | 10 | 0 | 12 | 32 |
| **D** | 50 | 0 | 12 | 0 | 15 |
| **E** | 0 | 5июня | 32 | 15 | 0 |

Логистический отдел службы доставки X хочет определить, каким путем быстрее доставить посылки из города А в город Е. Проанализировав карту дорог страны, специалисты логистического отдела построили граф, изображенный на рис. 18.

Рисунок восемнадцатого) г раф, вершины которого города, ребра - дороги между ними, а веса ребер - длины дорог; в) матрица смежности приведенного графу

**1** **Шаг.** Присваиваем каждой вершине метки в. Первой - 0, другим ­ - ∞. Новое юемо массив L, в который будем записывать путь и от А до других вершин и массив Mark для значения меток и информации о зачеркнутые метки.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L = | **A** | **B** | **C** | **D** | **E** |
| A |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mark = | **A** | **B** | **C** | **D** | **E** |
| 0 | ∞ | ∞ | ∞ | ∞ |
|  | 1 | 1 | 1 | 1 | 1 |

Рисунок 9 января

**2** **Шаг.** Присвоим меткам вершин, находящихся на расстоянии одного перехода от вершины А, длины этих переходов, если они меньше текущие значения метки.Укажем новые маршруты т а зачеркнем вершину А.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L = | **A** | **B** | **C** | **D** | **E** |
| A | A | A | A |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mark = | **A** | **B** | **C** | **D** | **E** |
| 0 | 78 | 30 | 50 | ∞ |
|  | 0 | 1 | 1 | 1 | 1 |

Рисунок 20

**3** **Шаг.** Выберем незакреслену вершину с наименьшей меткой. С ейчас это вершина С. присвоил меткам вершин, находящихся на расстоянии одного перехода от вершины С, длины этих переходов + Значение метки С, если ц и значение менее и по текущим и значения соответствующих меток. Укажем новые маршруты изачеркнем вершину С.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L = | **A** | **B** | **C** | **D** | **E** |
| A | С | A | С | С |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mark = | **A** | **B** | **C** | **D** | **E** |
| 0 | 40 | 30 | 42 | 62 |
|  | 0 | 1 | 0 | 1 | 1 |

Рисунок 21

**4** **Шаг.** Выберем незакреслену вершину с наименьшей меткой. Сейчас это вершина В. Присвоим меткам вершин, находящихся на расстоянии одного перехода от вершины В, длины этих переходов + значение метки, если эти значения меньше текущие значения соответствующих меток. Новых маршрутов нет. С акреслимо вершину В.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L = | **A** | **B** | **C** | **D** | **E** |
| A | С | A | С | С |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mark = | **A** | **B** | **C** | **D** | **E** |
| 0 | 40 | 30 | 42 | 62 |
|  | 0 | 0 | 0 | 1 | 1 |

Рисунок 22

**5** **Шаг.** Выберем незакреслену вершину с наименьшей меткой. Сейчас это вершина D. Присвоим меткам вершин, находящихся на расстоянии одного перехода от вершины D, длины этих переходов + значение метки D, если эти значения меньше текущие значения соответствующих меток. Укажем новые маршруты и зачеркнем вершину D.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L = | **A** | **B** | **C** | **D** | **E** |
| A | С | A | С | D |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mark = | **A** | **B** | **C** | **D** | **E** |
| 0 | 40 | 30 | 42 | 57 |
|  | 0 | 0 | 0 | 0 | 1 |

Рисунок 23

**Результат:** Матрица L указывает в город через которое ехать. Из нее можно в обратном порядке проследить кратчайший маршрут с любую ого городов а в город А. Кратчайший путь от города Е в города а А лежит через города D и С. Итак, с чения необходимого маршрута: A → C → D → Е.

**Задания** **к лабораторной работе:**

Реализовать программно абстрактный тип данных неориентированный граф. Создать его экземпляр, из аповниты данными, указанными в Вашем варианте.Разработать программу, которая должна:

1.   Построить для данного графу минимальное каркасное дерево, ис овуючы алгоритм Прима-Крускала, вы с начиты суммарный вес построенного дерева,вывести полученные данные на экран.

2.   Определить и вывести на экран кратчайший путь от вершины 1 к вершине 6 с помощью алгоритма Дейкстры.

вариант 1

вариант 2

вариант 3

вариант 4

вариант 5

вариант 6

вариант 7

вариант 8

вариант 9

вариант 10

вариант 11

вариант 12

вариант 13

вариант 14

вариант 15