Дискретная математика

Лабораторная работа 11
Кратчайшие пути в графах (4 часа)

Задание 1. Дан взвешенный граф, содержащий n вершин и m ребер. Веса всех ребер неотрицательны. Найти в заданном графе кратчайший путь из вершины s в вершину t с помощью алгоритма Дейкстры за $O(n^2)$. Вывести длину кратчайшего пути и номера вершин на пути от s до t.

Задание 2. Дан взвешенный граф, содержащий n вершин и m ребер. Веса всех ребер неотрицательны. Найти в заданном графе длины кратчайших путей между всеми парами вершин с помощью алгоритма Флойда за $O(n^3)$. Вывести матрицу кратчайших путей. Для пары вершин s и t вывести длину кратчайшего пути и номера вершин на пути от s до t.

Задание 3. Дан взвешенный граф, содержащий n вершин и m ребер. Веса ребер могут быть отрицательными. Найти в заданном графе кратчайший путь из вершины s в вершину t с помощью алгоритма Форда-Беллмана за $O(n^3)$. Вывести длину кратчайшего пути и номера вершин на пути от s до t или в случае существования в графе цикла отрицательного веса вывести соответствующее сообщение.

Алгоритм Дейкстры ($O(n^2)$)

Массив `d[]` – массив кратчайших расстояний от вершины `s` до всех остальных (`d[k]` – кратчайшее расстояние от вершины `s` до вершины `k`). Перед началом работы алгоритма установить `d[s] = 0`, для всех остальных вершин `d[v] = inf`.

Массив `u[]` – массив пометок уже посещенных вершин графа. Перед началом алгоритма установить для всех вершин `u[v] = false`.

Массив `p[]` – массив номеров предыдущих вершин (`p[k]` – номер вершины, из которой произошло улучшение пути до вершины `k`). Перед началом алгоритма установить для всех вершин `p[v] = -1`.

Алгоритм Дейкстры состоит из `n` итераций.

На каждой итерации:

- выбирается еще не посещенная вершина `v`, до которой расстояние от `s` минимально
- происходит, если это возможно, улучшение длины пути до еще не посещенных смежных с `v` вершин `to`:
  - `len` – длина ребра `(v,to)`
  - если **`d[v] + len < d[to]`**, то найденный путь до вершины to через вершину v короче ранее найденных, значит, **`d[to] = d[v] + len`** и **`p[to] = v`**
- выбранная вершина отмечается как посещенная

Для восстановления пути нужно вывести: `s ... p[p[p[t]]] p[p[t]] p[t] t`.

Алгоритм Флойда ($O(n^3)$)

Матрица `d[][]` – матрица длин кратчайших путей между всеми парами вершин. До начала работы алгоритма `d[i][j]` равно длине ребра из вершины `i` в вершину `j` или `inf` в случае отсутствия ребра.

Алгоритм Флойда состоит из `n` итераций.

На `k`-ой итерации:

- `d[i][j]` – длина кратчайшего пути из вершины `i` в вершину `j` при условии, что внутренними вершинами пути могут быть только вершины с номерами `1...k-1`
- для каждой пары вершин `i` и `j` происходит улучшение длины пути через вершину `k`: если **`d[i][k] + d[k][j] < d[i][j]`**, то **`d[i][j] = d[i][k] + d[k][j]`**

Алгоритм Форда-Беллмана ($O(n^3)$)

Массив `d[]` – массив кратчайших расстояний от вершины `v` до всех остальных (`d[k]` – кратчайшее расстояние от вершины `v` до вершины `k`). Перед началом работы алгоритма установить `d[s] = 0`, для всех остальных вершин `d[v] = inf`.

Алгоритм Форда-Беллмана состоит из `n-1` итерации, на каждой из которых просматриваются все ребра `(v,to)` графа и происходит улучшение длины пути из `v` в `to`: если **`d[v] + len < d[to]`**, то **`d[to] = d[v] + len`**, где `len` – длина ребра `(v,to)`.

Для обнаружения цикла отрицательного веса в графе необходимо выполнить еще одну итерацию. Если на ней произойдет хотя бы одно уменьшение значения длины пути, то граф содержит цикл отрицательного веса, в противном случае, такого цикла нет.

Важное замечание! В данных заданиях считаем, что на вход подаются взвешенные орграфы.