Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Российский химико-технологический университет имени Д.И. Менделеева»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

Выполнил студент группы КС-36: Золотухин А.А.
Ссылка на репозиторий: https://github.com/MUCTR-IKT-CPP/ZolotukhinAA_36_ALG
Принял: Крашенников Роман Сергеевич
Дата сдачи: 24.03.2025

Москва
2025

Оглавление

Описание задачи

В рамках лабораторной работы необходимо реализовать генератор случайных графов, генератор должен содержать следующие параметры:

- максимальное/минимальное количество генерируемых вершин;
- максимальное/минимальное количество генерируемых рёбер;
- максимальное количество рёбер, связанных с одной вершиной;
- генерируется ли направленный граф;
- максимальное количество входящих и выходящих рёбер.

Сгенерированный граф должен быть в рамках одного класса (этот класс не должен заниматься генерацией) и должен обладать обязательно следующими методами:

- выдача матрицы смежности;
- выдача матрицы инцидентности;
- выдача списка смежности;
- выдача списка рёбер.

В качестве проверки работоспособности требуется сгенерировать 10 графов с возрастающим количеством вершин и рёбер (количество выбирать в зависимости от сложности расчёта для вашего отдельно взятого ПК)

На каждом из сгенерированных графов требуется выполнить поиск кратчайшего пути или подтвердить его отсутствие из точки А в точку Б, выбирающиеся случайным образом заранее, поиском в ширину и поиском в глубину, замерев время, требуемое на выполнение операции. Результаты замеров наложить на график и проанализировать эффективность применения обоих методов к этой задаче.

Описание метода/модели

Вершина графа - некоторая точка, связанная с другими точками.

Ребро графа - линия, соединяющая две точки и олицетворяющая связь между ними.

Граф - множество вершин, соединённых друг с другом произвольным образом множеством рёбер.

Ориентированным графом называют такой граф, в котором каждое ребро имеет направление движения, и, как правило, не преполагает возможности обратного перемещения.

Описание графа. Для описания графа используют один из следующих вариантов:

- **Матрица смежности** - двумерная таблица, для которой столбцы соответствуют вершинам, а значения в таблице соответствуют рёбрам, для *невзвешенного* графа они могут быть просто 1, если связь есть и идёт в нужном направлении, и 0, если её нет, а для *невзвешенного* графа будут стоять конкретные значения.

- **Матрица инцидентности** - матрица, в которой строки соответствуют вершинам, а столбцы соответствуют связям, и в ячейке ставится 1, если связь выходит из вершины, -1, если входит, и 0 во всех остальных случаях.

- **Список смежности** - список списков, содержащий все вершины, а внутренние списки для каждой вершины содержат все смежные ей.

- **Список рёбер** - список строк, в которых хранятся все рёбра вершины, а внутреннее значение содержит две вершины, к котором присоединено это ребро.

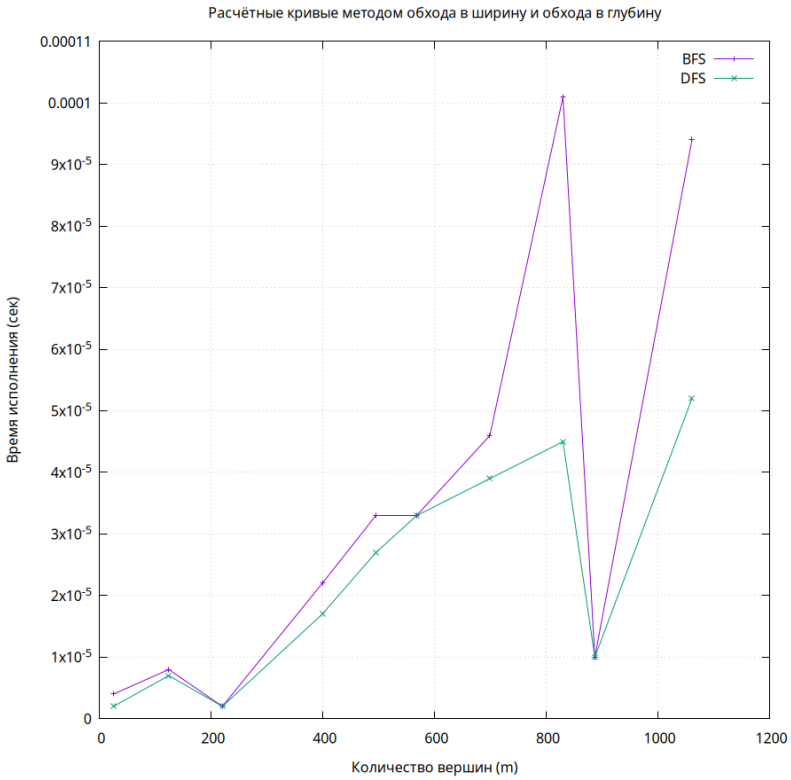
Обход графа - переход от одной его вершины к другой в поисках свойств связей этих вершин. Выделяют два варианта обхода: обход в глубину и обход в ширину.

DFS (deep first search) следует концепции "погружайся глубже, в головой вперёд". Идея заключается в том, что мы двигаемся от начальной вершины в определённом направлении до тех пор, пока не достигнем конца пути. Если мы достигли конца пути, но он не является пунктом назначения, то мы возвращаемся назад (к точке разветвления) и идём по другому маршруту.

BFS (breadth first search) следует концепции "расширяйся, поднимаясь на высоту птичьего полёта". Вместо того, чтобы двигаться по определённом пути до конца, BFS предполагает движение вперёд по одному соседу за раз, вместо следования по пути, BFS подразумевает посещение ближайших к вершине соседей за одно действие, затем посещение соседей соседей и так до тех пор, пока не будет обнаружена исковая вершина или соседи не закончатся.

Выполнение задачи

Алгоритмы обхода в ширину и обхода в глубину реализованы на языке *C++*. Построение графиков с помощью программы *GNUplot*. Демонстрация графов была сделана с помощью *Graphviz*.


Расчётные кривые методом обхода в ширину и обхода в глубину
BFS
DFS
Время исполнения (сек)
Количество вершин (m)
0.00011
0.0001
9x10-5
8x10-5
7x10-5
6x10-5
5x10-5
4x10-5
3x10-5
2x10-5
1x10-5
0
0
200
400
600
800
1000
1200

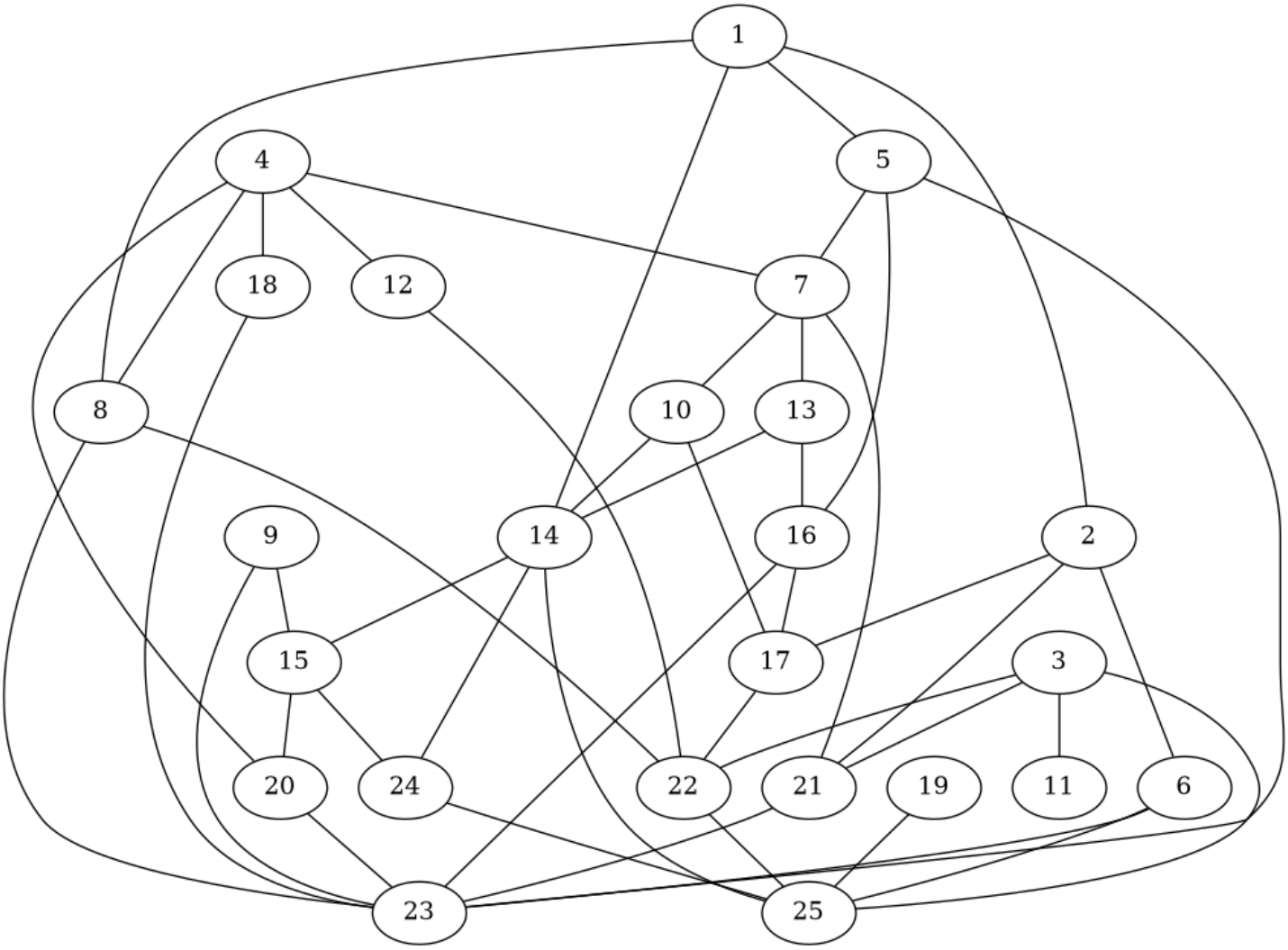


Рис. 1: Граф 1.

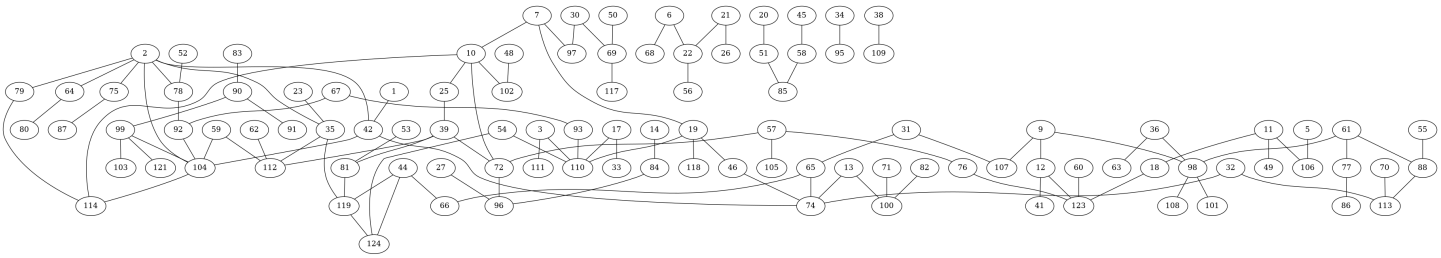


Рис. 2: Граф 2.

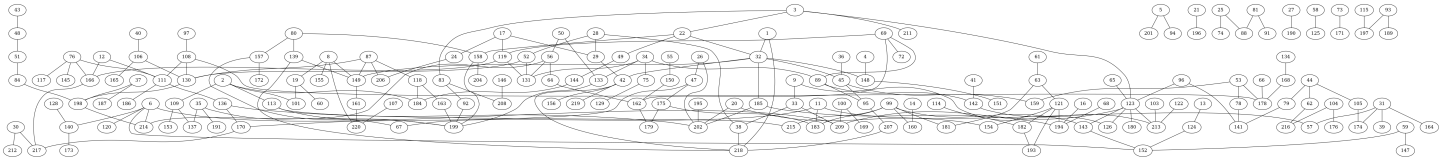


Рис. 3: Граф 3.

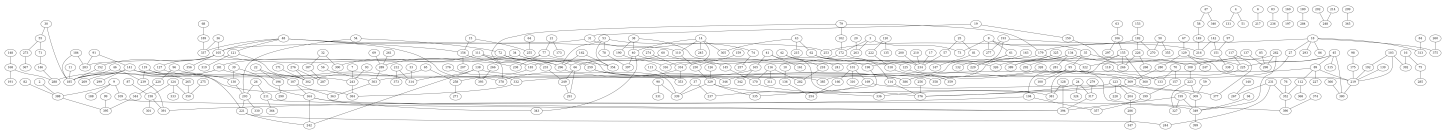


Рис. 4: Граф 4.

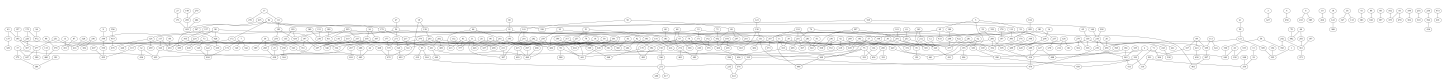


Рис. 5: Граф 5.

Выводы

В ходе лабораторной работы я изучил, что такое граф, разновидности графов, методы обхода графов. Алгоритм обхода в глубину работает быстрее по времени, чем алгоритм обхода в ширину, поскольку BFS ищет кратчайший путь, а DFS - любой возможный.