УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ   
УНИВЕРСИТЕТ»

Отчет по лабораторной работе №5

«АЛГОРИТМЫ НА ГРАФАХ»

(Вариант 12)

Выполнила: Мергель К.А.

ФИТ, 2 к., 4 гр.

Проверила: Харланович А. В.

Минск 2020

Целью работы было освоить сущность и программную реализацию: а) способов представления графов; б) алгоритмов поиска в ширину и глубину; в) алгоритма топологической сортировки графов

Задание 1:

Необходимо было осуществить алгоритмы поиска в ширину и глубину, а также алгоритма топологической сортировки аналогично примерам, рассмотренным на лекциях.



Алгоритм поиска в глубину

Для обозначения алгоритма поиска в глубину используют аббревиатуру ***DFS*** (***Depth-first search***). Задается стартовая вершина. Алгоритм описывается следующим образом: для каждой не пройденной вершины, начиная со стартовой, необходимо найти все смежные вершины и повторить поиск для каждой.

В основе алгоритма DFS лежит рекурсивная процедура **Visit**, имеющая один входной параметр **k** – вершину графа.

Опишем пошагово процедуру **Visit**.

1. Принять параметр **k** – вершину графа.
2. Вершину **k**  окрасить в серый цвет: **C[k] = G**.
3. Увеличить номер шага: **t = t + 1**.
4. Подсчитать расстояние до вершины: **D[k] = t**. Расстояние до вершины в алгоритме DFS совпадает с номером шага, на котором эта вершина была обнаружена (окрашена в серый цвет).
5. Построить множества **J** вершин белого цвета, смежных вершине **k**. Если таких вершин нет, то перейти к шагу 8.
6. Для каждой вершины **j** из множества **J** указать предшествующую вершину: **P[j] = k**.
7. Для каждой вершины **j** из множества **J** выполнить процедуру **Visit**.
8. Вершину **k**  окрасить в черный цвет: **C[k] = B**.
9. Увеличить номер шага: **t = t + 1**.
10. Отметить время фиксации вершины: **F[k] = t**.

Если задана стартовая вершина **s**, то алгоритм DFS теперь можно свести к следующим двум шагам.

1. Инициализировать массивы: **С** (все вершины белого цвета), **D** (все расстояния равны бесконечности), **P** (все элементы заполнены символом «пустота»). Установить номер шага: **t = 0**.
2. Выполнить процедуру **Visit**  для вершины **s**.



Изначально все вершины белые, а стек пуст. Переходи к вершине 0. Красим её в серый.

С-массив окраски вершины, Р-массив предшествующих вершин. D-время обнаружения (шаг окраски в серый цвет), F-хранения времени фиксации, t- номер шага алгоритма

t=1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | W | W | W | W | W | W |
| D | 1 | I | I | I | I | I | I |
| P | N | N | N | N | N | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Ищем смежные.

Затем переходим в 1 и красим её тоже в серый цвет



t=2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | G | W | W | W | W | W |
| D | 1 | 2 | I | I | I | I | I |
| P | N | 0 | N | N | N | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Из 1 не можем никуда перейти, поэтом возвращаемся в 0 и переходим 3.



t=3, t=4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | W | G | W | W | W |
| D | 1 | 2 | I | 4 | I | I | I |
| P | N | 0 | N | 0 | N | N | N |
| F | 0 | 3 | 0 | 0 | 0 | 0 | 0 |

Затем переходим в 2 и красим тоже в серый цвет



t=5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | W | W | W |
| D | 1 | 2 | 5 | 4 | I | I | I |
| P | N | 0 | 3 | 0 | N | N | N |
| F | 0 | 3 | 0 | 0 | 0 | 0 | 0 |

Затем переходим в 4 и красим тоже в серый цвет



t=6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | W | G | W |
| D | 1 | 2 | 5 | 4 | I | 6 | I |
| P | N | 0 | 3 | 0 | 3 | N | N |
| F | 0 | 3 | 0 | 0 | 0 | 0 | 0 |



t=7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | W | G | G |
| D | 1 | 2 | 5 | 4 | I | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | N | 5 |
| F | 0 | 3 | 0 | 0 | 0 | 0 | 0 |



t=8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | G | G | G |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 0 | 0 | 0 | 0 | 0 |

Белых вершин, смежный с 5, нет, следовательно, мы погрузились в глубину. Закрашиваем 5 вершину в чёрный цвет.



t=9

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | B | G | G |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 0 | 0 | 9 | 0 | 0 |



t=10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | B | G | B |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 0 | 0 | 9 | 0 | 10 |



t=11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | G | G | B | B | B |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 0 | 0 | 9 | 11 | 10 |



t=12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | B | G | B | B | B |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 12 | 0 | 11 | 10 | 9 |



t=13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | G | B | B | B | B | B | B |
| D | 1 | 2 | 5 | 4 | 8 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 0 | 3 | 12 | 13 | 9 | 11 | 10 |



t=14

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| C | B | B | B | B | B | B | B |
| D | 1 | 2 | 5 | 4 | 11 | 6 | 7 |
| P | N | 0 | 3 | 0 | 3 | 2 | 5 |
| F | 14 | 3 | 10 | 13 | 12 | 9 | 8 |

Алгоритм поиска в ширину

Алгоритм подразумевает, что задана исходная (***стартовой***) вершина, и основывается на простом правиле: при выборе очередной вершины предпочтение отдается ближайшей. При этом считается, что все дуги графа имеют единичную длину. Сначала посещается стартовая вершина, затем все вершины, смежные ей (т. е. находящиеся на расстоянии 1), после чего вершины, находящиеся на расстоянии 2 от стартовой и т.д.

Алгоритм BFS сводится к следующей последовательности шагов.

1. Инициализировать массивы С, D, P. Стартовую вершину s поместить в очередь Q. и окрасить в серый цвет: C[s] = G. Для стартовой вершины установить расстояние, равное нулю: D[s] = 0.
2. Если очередь Q пуста, то работа алгоритма завершена, в противном случае перейти к следующему шагу.
3. Выбрать из очереди Q вершину k и окрасить ее в черный цвет: С[k] = B.
4. Построить множества J вершин белого цвета смежных вершине k. Если таких вершин нет, то перейти к шагу 2, иначе – к следующему шагу.
5. Каждую вершину j из множества J поместить в очередь Q. Обычно (но не обязательно) в очередь вершины помещаются в порядке возрастания номеров.
6. Каждую вершину j из множества J окрасить в серый цвет: С[j] = G.
7. Для каждой вершины j из множества J вычислить расстояние: D[j] = D[k] + 1.
8. Для каждой вершины j из множества J указать предшествующую вершину: P[j] = k.
9. Перейти к шагу 3.



Q-очередь вершин, С-массив окраски, D-массив расстояний, Р-массив предшествующих вершин.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 0 |  |  |  |  |  |  |
| C | G | W | W | W | W | W | W |
| D | 0 | I | I | I | I | I | I |
| P | N | N | N | N | N | N | N |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 1 | 3 |  |  |  |  |  |
| C | B | G | W | W | W | W | W |
| D | 0 | 1 | I | 1 | I | I | I |
| P | N | 0 | N | 0 | N | N | N |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 3 |  |  |  |  |  |  |
| C | B | B | W | G | W | W | W |
| D | 0 | 1 | I | 1 | I | I | I |
| P | N | 0 | N | 0 | N | N | N |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 2 | 4 | 6 |  |  |  |  |
| C | B | B | G | B | G | W | G |
| D | 0 | 1 | 2 | 1 | 2 | I | 2 |
| P | N | 0 | 3 | 0 | 3 | N | 3 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 4 | 6 | 5 |  |  |  |  |
| C | B | B | B | B | G | G | G |
| D | 0 | 1 | 2 | 1 | 2 | 3 | 2 |
| P | N | 0 | 3 | 0 | 3 | 2 | 3 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 6 | 5 |  |  |  |  |  |
| C | B | B | B | B | B | G | G |
| D | 0 | 1 | 2 | 1 | 2 | 3 | 2 |
| P | N | 0 | 3 | 0 | 3 | 2 | 3 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 5 |  |  |  |  |  |  |
| C | B | B | B | B | B | B | G |
| D | 0 | 1 | 2 | 1 | 2 | 3 | 2 |
| P | N | 0 | 3 | 0 | 3 | 2 | 3 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q |  |  |  |  |  |  |  |
| C | B | B | B | B | B | B | B |
| D | 0 | 1 | 2 | 1 | 2 | 3 | 2 |
| P | N | 0 | 3 | 0 | 3 | 2 | 3 |

Все вершины закрашены, следовательно, поиск в ширину окончен.

**Алгоритм топологической сортировки**

1/



2/

1/



2/3

1/



4/

|  |
| --- |
| **1** |

1/

2/3



7/

6/

5/

4/



1/14

11/12

4/13

6/9

7/8

5/10

2/3

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

Формируется очередь по мере окрашивания в черный цвет: 4 3 2 1 0.

0

**X`**

Задания 2-4:

Необходимо было осуществить программную реализацию алгоритмов на C++. Разработать структуры **AMatrix** и **АList**  для представления ориентированного графа матричным и списковым способом; разработать функции преобразования из одного способа представления в другой; разработать функцию **BFS** обхода вершин графа, используя метод поиска в ширину; разработать функцию **DFS**  обхода вершин графа, используя метод поиска глубину; доработать функцию **DFS**,для выполнения топологической сортировки графа.

***Топологическая сортировка*** − это процедура упорядочивания вершин ориентированного графа, не имеющего циклов (ациклического графа). В результате топологической сортировки для вершин графа определяется такой порядок, что если их расположить на рисунке в соответствии с этим порядком сверху вниз, то дуги будут направлены только от верхних вершин к нижним. Обычно после выполнения топологической сортировки вершины переименовываются (перенумеровываются) в соответствии с полученным порядком. После такого переименования граф обладает свойством: начальная вершина каждой дуги имеет номер (имя) меньший, чем номер конечной вершина этой дуги.

Структура AMatrix содержит количество вершин, матрицу, функции создания нулевой матрицы n\*n, создания матрицы n\*n, создания подобной матрицы, создания матрицы из спискового, записи mr[i, j] = r, получения элемента mr[i, j].

Структура AList содержит количество вершин, массив списков, функции создания массива пустых списков, спискового представления, подобной структуры, добавления в i-ый список, получения размера i-го списка, j-ого элемента i-го списка.

Функция BFS находится в одноименной структуре, которая содержит перечисление цветов, исходный граф, цвет вершины, расстояние до вершины, предшествующую вершину, очередь, саму функцию в двух вариантах (для представления матричным и списковым способами) и функцию получения следующей вершины. Структура имеет вид:

|  |
| --- |
| struct BFS  {  const static int INF = 0x7fffffff;  const static int NIL = -1;  enum Color { WHITE, GRAY, BLACK };  const graph::AList \*al;  Color \*c;  int \*d;  int \*p;  std::queue<int> q;  BFS(const graph::AList& al, int s);  BFS(const graph::AMatrix& am, int s);  void init(const graph::AList& al, int s);  int get();  }; |

Функция DFS находится в одноименной структуре, которая содержит перечисление цветов, исходный граф, цвет вершины, время обнаруженія, время завершения обработки, предшествующую вершину, текущее время, саму функцию в двух вариантах (для представления матричным и списковым способами), результат топологической сортировки, функцию получения вершины. Структура имеет вид:

|  |
| --- |
| struct DFS  {  const static int NIL = -1;  enum Color { WHITE, GRAY, BLACK };  const graph::AList \*al;  Color \*c;  int \*d;  int \*f;  int \*p;  int t;  DFS(const graph::AList& al);  DFS(const graph::AMatrix& am);  std::vector <int> topological\_sort;  void visit(int v);  void init(const graph::AList& al);  int get(int i);  }; |

Результат работы функций представлен ниже:

