Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ, Информатика и системы управления

КАФЕДРА ИУ7, Программное обеспечение ЭВМ и информационные технологии

ОТЧЕТ ПО ДОМАШНЕМУ ЗАДАНИЮ №1

ПО ДИСЦИПЛИНЕ

“Анализ алгоритмов”

Студент ИУ7-54Б _____ **А.А. Андреев**
(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель _____ **Л.Л. Волкова**
(Подпись, дата) (И.О.Фамилия)

2021 г.

Оглавление

Введение.

Данная домашняя работа посвящена построению графовых моделей на основе кода.

Цель данной лабораторной работы: Изучение и реализация графовых моделей: графа управления, операционной истории, информационного графа, информационной истории.

Задачи данной лабораторной работы:

1. Изучение понятие графовых моделей;
2. Реализовать фрагмент программы построения методом наименьших квадратов полинома, аппроксимирующего функцию от двух переменных, заданную таблицей с весами (в частности, фрагмент кода, необходимый для расчета коэффициентов аппроксимации);
3. На основе реализованного фрагмента кода построить граф управления, операционную историю, информационный граф, информационную историю;

1. Аналитическая часть

Графовая модель [1] - математическая абстракция реальной системы любой природы, объекты которой обладают парными связями. Граф как математический объект есть совокупность двух множеств — множества самих объектов, называемого множеством вершин, и множества их парных связей, называемого множеством рёбер.

В графе потока управления каждый узел (вершина) графа соответствует базовому блоку — прямолинейному участку кода, не содержащему в себе ни операций передачи управления, ни точек, на которые управление передается из других частей программы. Имеется лишь два исключения:

- точка, на которую выполняется переход, является первой инструкцией в базовом блоке;
- базовый блок завершается инструкцией перехода.

Направленные дуги используются в графе для представления инструкций перехода. Также, в большинстве реализаций добавлено два специализированных блока:

- **входной блок**, через который управление входит в граф;
- **выходной блок**, который завершает все пути в данном графе.

Структура CFG важна для многих оптимизаций компиляторов и для утилит статического анализа кода.

Возможны два случая: у блока или подграфа отсутствует:

- входной блок («мёртвый» код);
- выходной блок (бесконечный цикл).

Блок, не связанный со входным блоком, считается недостижимым («мёртвый» код).

Достижимость — одно из свойств графа, используемое при оптимизациях. Недостижимый блок может быть удален из программы.

Блок, не связанный с выходным блоком, содержит бесконечный цикл. Полагаясь на это высказывание, удаётся обнаружить не все бесконечные циклы из-за проблемы остановки.

При выполнении оптимизаций компилятор может создавать и «мёртвый» код, и бесконечные циклы, даже если программист явно это не кодировал. Например, после выполнения свертки констант (англ. *constant folding*) и распространения констант (англ. *constant propagation*) оптимизация jump threading может соединить несколько блоков в один; в результате некоторые

ребра могут исчезнуть и некоторые блоки могут оказаться не связанными с графом.

Вывод

В данном разделе было рассмотрено понятие графовой модели, обозначены схемы графа управления, операционной истории, информационного графа, информационной истории.

2. Технологическая часть.

В данном разделе будут рассмотрены требования к разрабатываемому программному обеспечению, средства, использованные в процессе разработки для реализации поставленных задач.

2.1 Требования к программному обеспечению

Программное обеспечение должно реализовывать поставленную на домашнюю работу задачу. Интерфейс для взаимодействия с программой - командная строка. Программа должна выводить определитель. Это программа построения методом наименьших квадратов полинома, аппроксимирующего функцию от двух переменных, заданную таблицей с весами (в частности, фрагмент, необходимый для расчета коэффициентов аппроксимации).

2.2 Выбор и обоснование языка и среды программирования.

Для разработки данной программы применён язык Python, чтобы расширить знания в области данного языка программирования.

2.3 Реализация алгоритмов

В листинге 1 приведена реализация фрагмента программы построения методом наименьших квадратов полинома, аппроксимирующих функцию от двух переменных, заданную таблицей с весами.

Листинг 1: Фрагмента программы построения методом наименьших квадратов полинома, аппроксимирующих функцию от двух переменных, заданную таблицей с весами

```
x, y, p, n = list(map(float, input("X: ").split())), \
                 list(map(float, input("X: ").split())), \
                 list(map(float, input("X: ").split())), \
                 int(input("Введите степень аппроксимации: "))
    N = len(x)
    A = zeros((n + 1, n + 1))
    B = zeros((n + 1, 1))
    for j in range(n + 1):
        for k in range(n + 1):
                for i in range(N):
                    A[j, k] += f(x[i], k + j) * p[i]
           for l in range(N):
               B[j] += y[l] * f(x[l], j) * p[l]
      det = linalg.det(A)
      nparray_to_list(A)
      nparray_to_list(b)
      print("Определитель:", det)
```

3.1 Графовые модели

3.1.1 Граф управления

Граф управления, где каждый пунсон - номер строк кода (см. Рисунок 1).

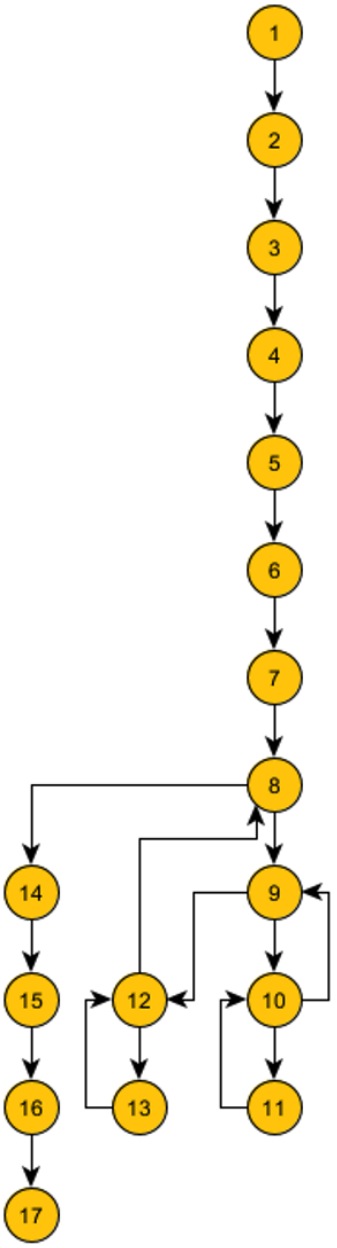

Рисунок 1: Граф управления

3.1.2 Операционная история

Операционная история, где каждый пунсон - номер строк кода (см. Рисунок 2).

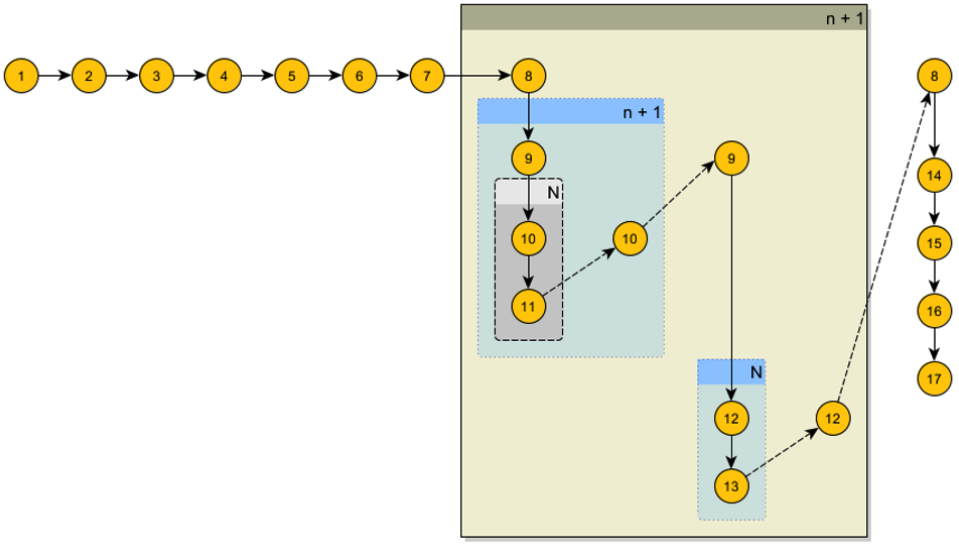

Рисунок 2: Операционная история

3.1.3 Информационный граф

Информационный граф, где каждый пунсон - номер строк кода (см. Рисунок 2).

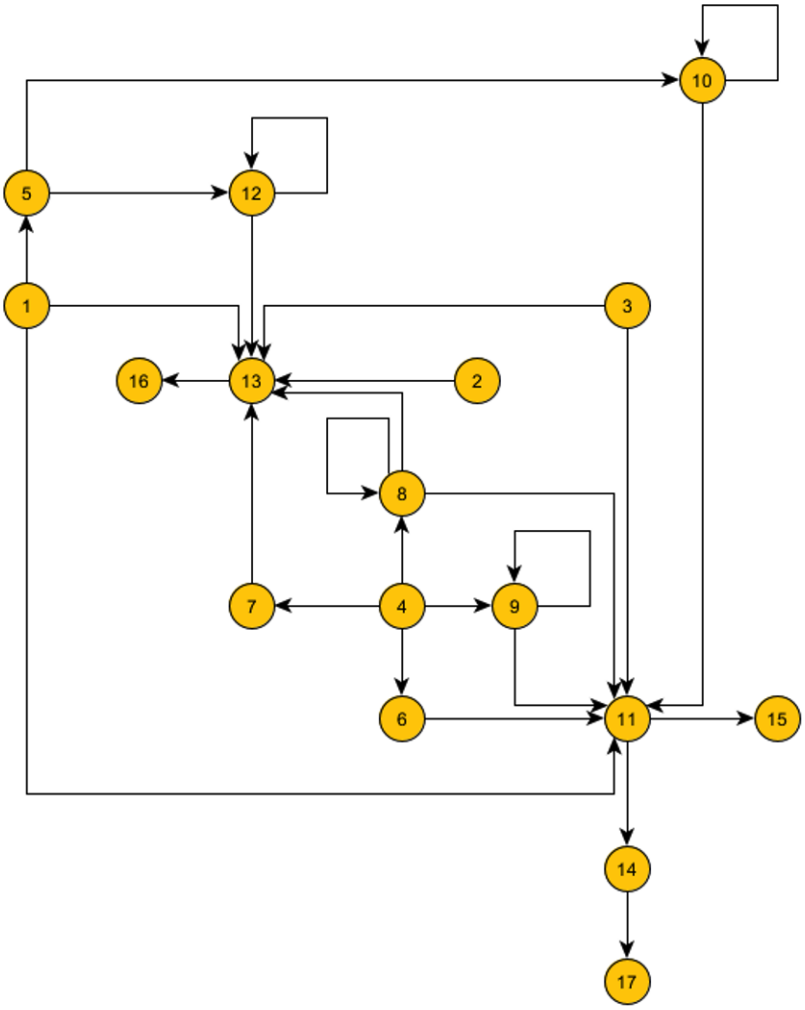

Рисунок 3: Информационный граф

3.1.4 Информационная история

Информационная история, где каждый пунсон - номер строк кода (см. Рисунок 2). Для очередных итераций внешнего цикла (вершина 8) в модели не были указаны дуги связей с целью сохранения наглядности графа (в частности, не указаны дуги связи для вершины 9, 10, 11, 12, 13, получаемые аналогично первой итерации из вершин 1, 2, 3, 4, 5, 6, 7)

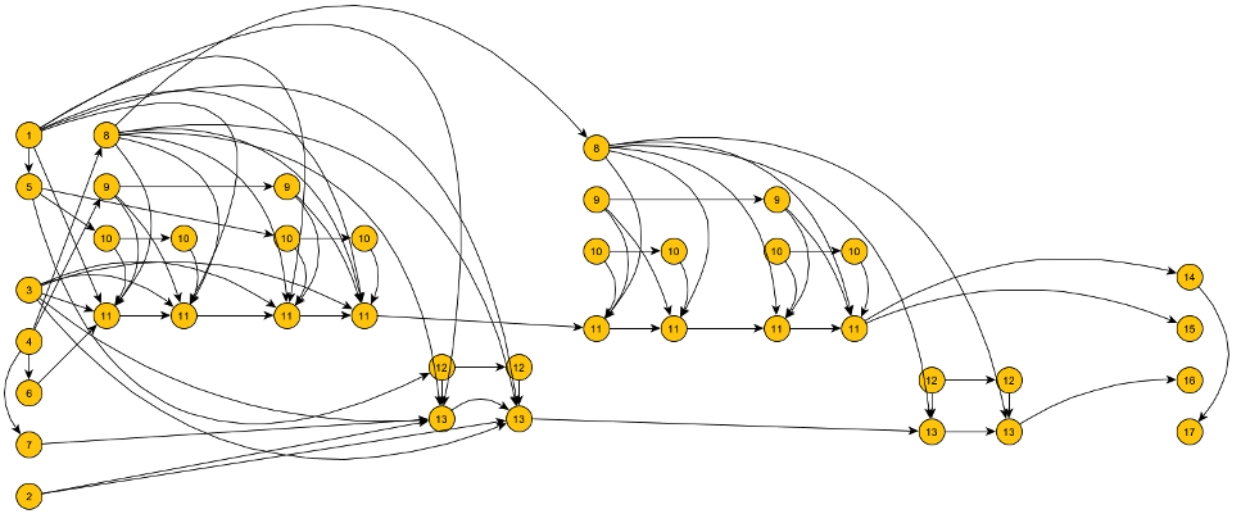

Рисунок 4: Информационная история

2.2 Вывод

В данном разделе были рассмотрены требования к разрабатываемому программному обеспечению, средства, использованные в процессе разработки для реализации поставленных задач, приведены результаты работы программы на тестовых данных.

Заключение.

В рамках данной домашней работы была достигнута ее цель: Изучение и реализация графовых моделей: графа управления, операционной истории, информационного графа, информационной истории, также выполнены задачи:

- Изучено понятие графовых моделей;
- Реализован фрагмент программы построения методом наименьших квадратов полинома, аппроксимирующего функцию от двух переменных, заданную таблицей с весами (в частности, фрагмент кода, необходимый для расчета коэффициентов аппроксимации);
- На основе реализованного фрагмента кода построены граф управления, операционную историю, информационный граф, информационную историю;

Список использованной литературы

[1] Гасфилд, Строки, деревья и последовательности в алгоритмах. Информатика и вычислительная биология. Невский Диалект БВХ-Петербург, год выпуска 2003, типаж 900, 653 страницы.