



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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по домашняя работа № 1
по курсу «Анализ алгоритмов»
на тему: «Трудоёмкость сортировок»

Студент ИУ7-54Б
(Группа)

(Подпись, дата)

Булдаков М.
(И. О. Фамилия)

Преподаватель

(Подпись, дата)

Волкова Л. Л.
(И. О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Аналитический раздел	4
1.1 Графовые модели программы	4

ВВЕДЕНИЕ

Цель данной лабораторной работы — описать четырьмя графовыми моделями (графом управления, информационным графом, операционной историей, информационной историей) последовательный алгоритм либо фрагмент алгоритма, содержащий от 15 значащих строк кода и от двух циклов, один из которых является вложенным в другой.

В качестве реализуемого алгоритма — вычисление расстояния Левенштейна.

1 Аналитический раздел

В данном разделе будет представлена информация о графовых моделях и приведены графовые модели поставленной задачи.

1.1 Графовые модели программы

Программа представлена в виде графа: набор вершин и множество соединяющих их направленных дуг.

Выделяют 2 типа дуг:

- 1) операционное отношение — по передаче управления;
- 2) информационное отношение — по передаче данных.

Граф управления — модель, в который **вершины** — операторы, **дуги** — операционные отношения.

Информационный граф — модель, в которой **вершины**: операторы, **дуги** — информационные отношения.

Операционная история — модель, в которой **вершины**: срабатывание операторов, **дуги** — операционные отношения.

Информационная история — модель, в которой **вершины**: срабатывание операторов, **дуги** — информационные отношения.

Графы более компактны, однако менее информативны, чем истории. Истории менее компактны, однако более информативны, чем графы.

На листинге 1.1 приведена реализации функции, вычисляющей расстояние Левенштейна.

Листинг 1.1 – Реализация алгоритма вычисления расстояния Левенштейна

```
1 def levenstein():
2     s1 = input("Введите 1-ую строку: ") # 1
3     s2 = input("Введите 2-ую строку: ") # 2
4
5     len1 = len(s1) # 3
6     len2 = len(s2) # 4
7
8     M = [[0] * (len2 + 1) for _ in range(len1 + 1)] # 5
9
10    for i in range(len1 + 1): # 6
11        M[i][0] = i # 7
12
13    for j in range(len2 + 1): # 8
14        M[0][j] = j # 9
15
16    for i in range(1, len1 + 1): # 10
17        for j in range(1, len2 + 1): # 11
18            A = M[i - 1][j] + 1 # 12
19            D = M[i][j - 1] + 1 # 13
20            C = M[i - 1][j - 1] # 14
21
22            if s1[i - 1] != s2[j - 1]: # 15
23                C += 1 # 16
24
25            M[i][j] = min(A, D, C) # 17
26
27    return M[-1][-1]
```

На рисунке 1.1 представлен граф управления. На рисунке 1.2 представлен информационный граф.

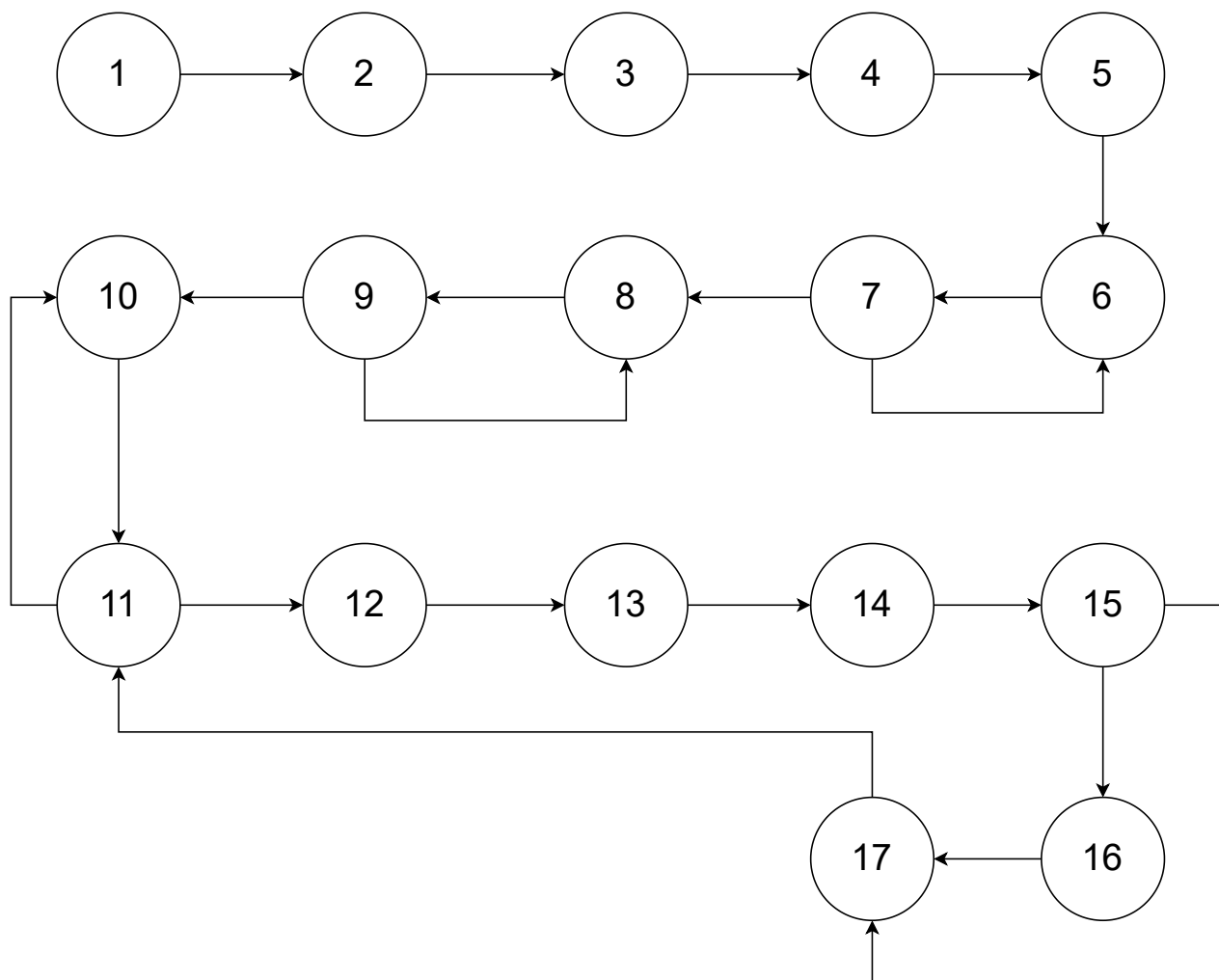


Рисунок 1.1 – Граф управления

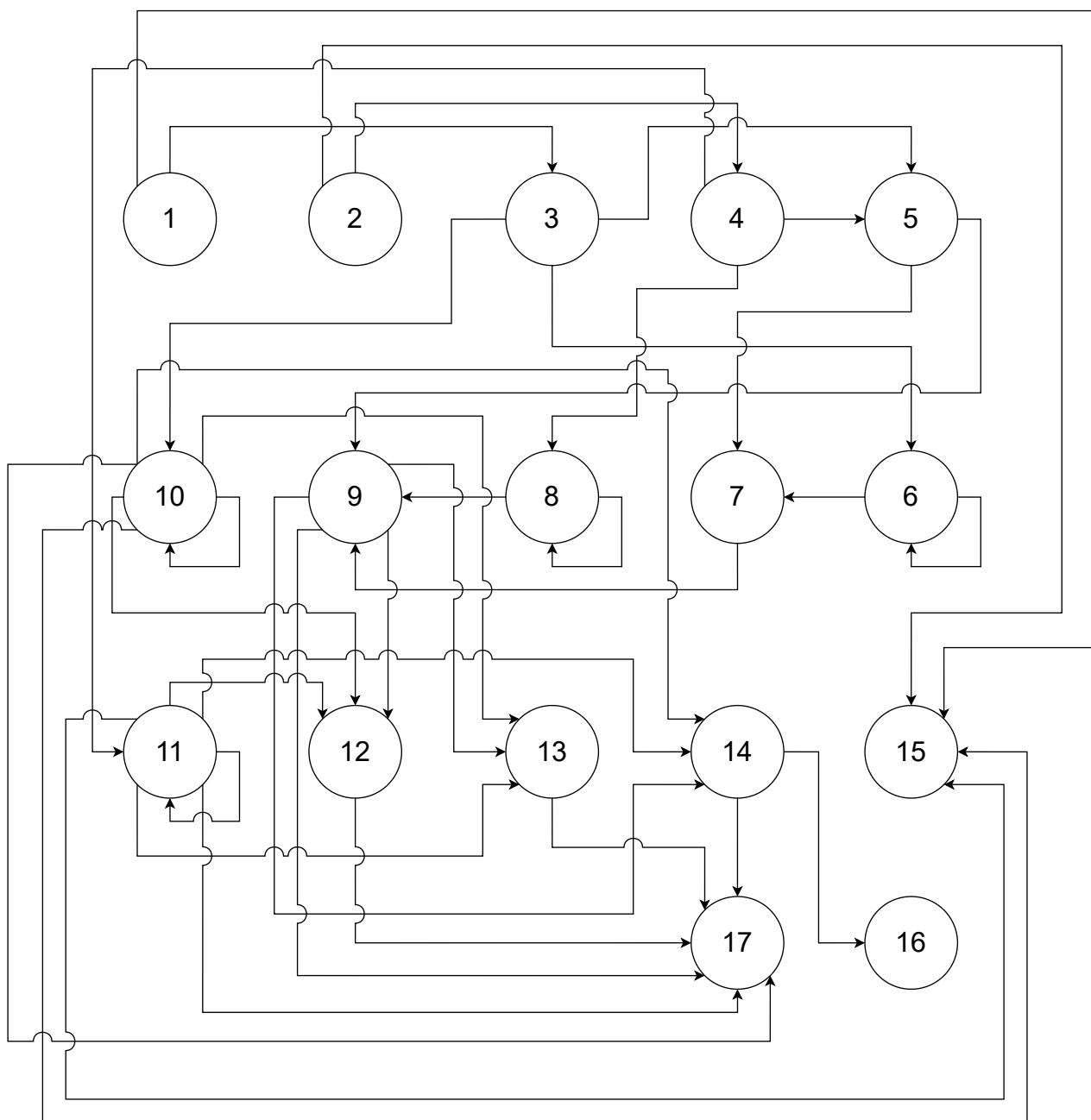


Рисунок 1.2 – Информационный граф

Приведем графы для строк «аба» и «aaa». На рисунке 1.3 представлена операционная история. На рисунке 1.4 представлена информационная история.

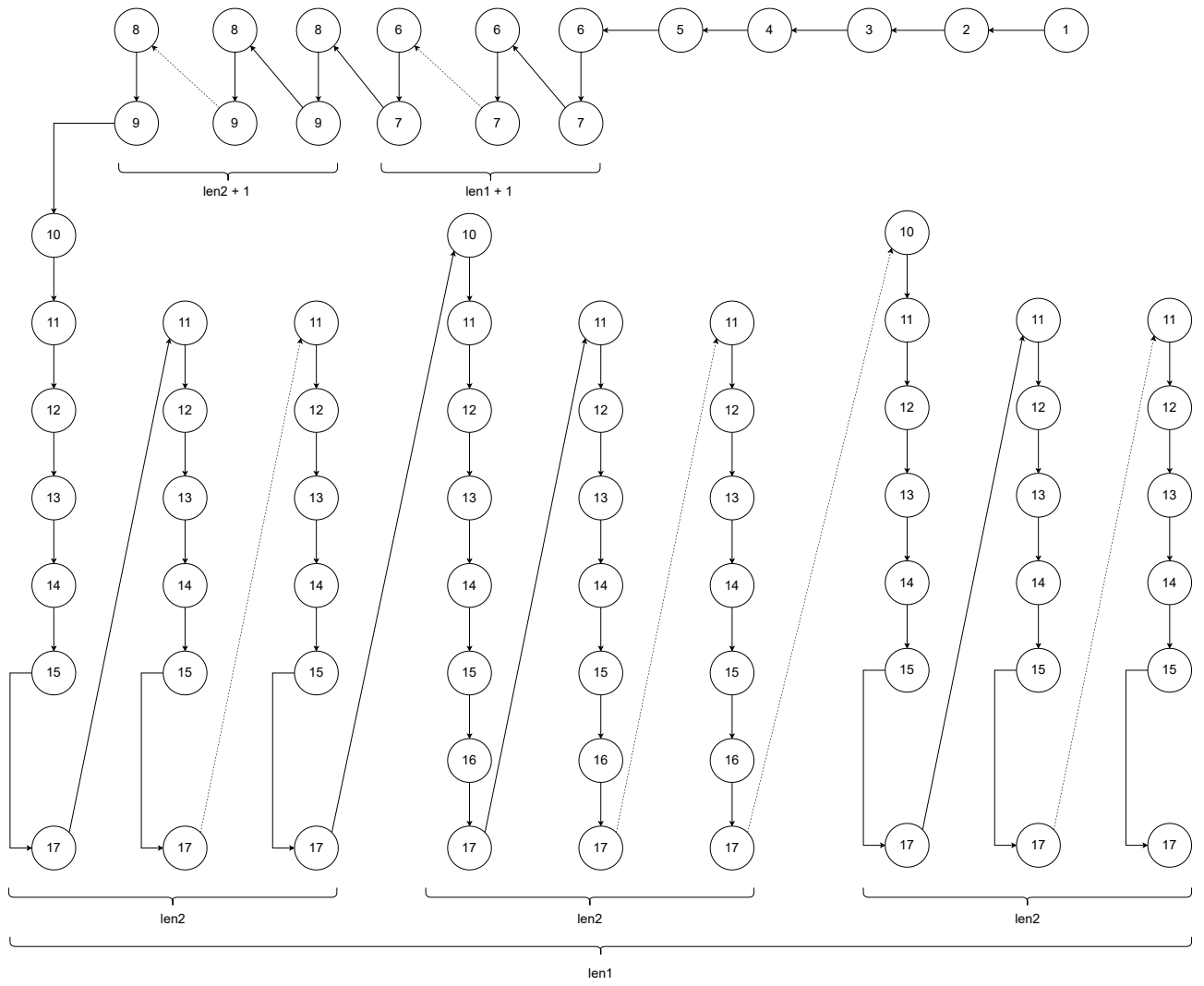


Рисунок 1.3 – Операционная история

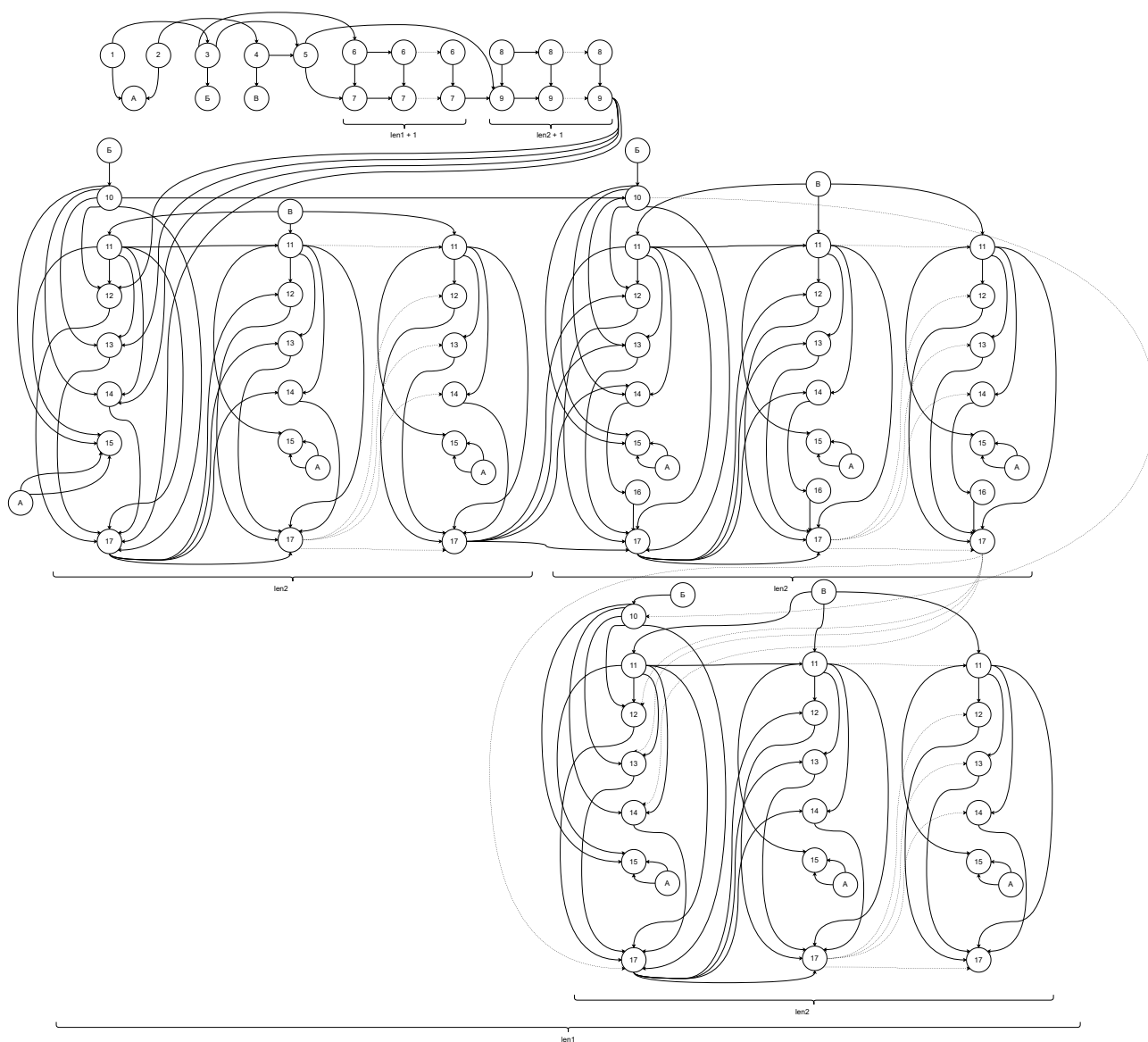


Рисунок 1.4 – Информационная история

Вывод

В реализуемом алгоритме наблюдается скошенный параллелизм. Можно вычислять диагонали, параллельные побочной в матрице Левенштейна, в различных потоках, поскольку элементы диагонали не зависят друг от друга.