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 домашнему заданию № 1

по курсу «Анализ алгоритмов»

на тему: «Параллельные вычисления»

Студент ИУ7-53Б _____ В. Марченко
(Группа) (Подпись, дата) (И. О. Фамилия)

Преподаватель _____ Ю. В. Строганов
(Подпись, дата) (И. О. Фамилия)

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

Москва — 2022 г.

СОДЕРЖАНИЕ

1 Реализация алгоритма

В листинге 1.1 показана реализация алгоритма слияния двух массивов для сортировки слиянием.

Листинг 1.1 – Реализация алгоритма слияния двух массивов

```
1  int merge(int *const array, const int begin_pos, const int
       middle_pos, const int end_pos)
2  {
3      int left_size = middle_pos - begin_pos + 1;           // 1
4      int right_size = end_pos - middle_pos;                // 2
5      int i = 0;                                            // 3
6      int j = 0;                                            // 4
7      int *left_array = malloc(left_size * sizeof(int));    // 5
8      int *right_array = malloc(right_size * sizeof(int)); // 6
9      for (i = 0; i < left_size; i++)
10         left_array[i] = array[begin_pos + i];             // 7
11     for (i = 0; i < right_size; i++)
12         right_array[i] = array[middle_pos + i + 1];       // 8
13     int k = begin_pos;                                    // 9
14     i = 0;                                                // 10
15     j = 0;                                                // 11
16     while (i < left_size && j < right_size)
17     {
18         if (left_array[i] <= right_array[j])
19         {
20             array[k] = left_array[i];                     // 12
21             k++;                                          // 13
22             i++;                                          // 14
23         }
24         else
25         {
26             array[k] = right_array[j];                    // 15
27             k++;                                          // 16
28             j++;                                          // 17
29         }
30     }
31     while (i < left_size)
32     {
33         array[k] = left_array[i];                         // 18
34         k++;                                              // 19
35         i++;                                              // 20
```

```
36      }
37      while (j < right_size)
38      {
39          array[k] = right_array[j];                    // 21
40          k++;                                          // 22
41          j++;                                          // 23
42      }
43      free(left_array);                                 // 24
44      free(right_array);
45      return 0;
46  }
```

2 Операционный граф

На рисунке 2.1 показан операционный граф.

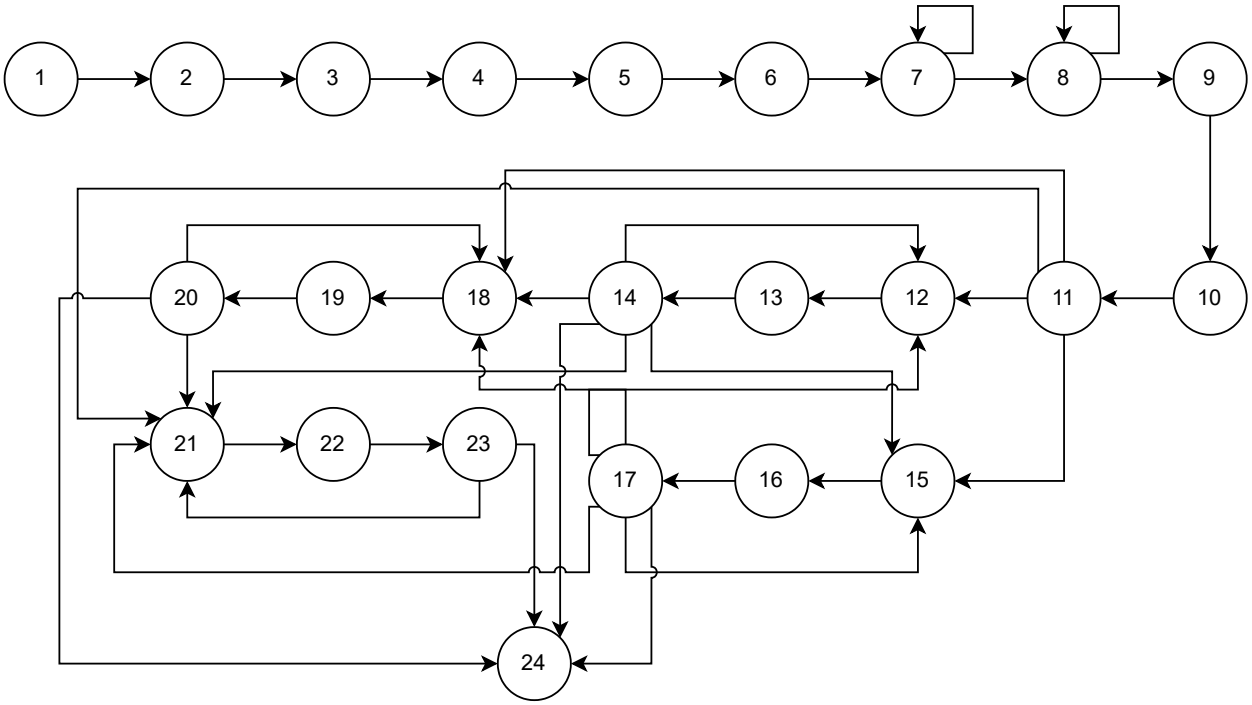

Рисунок 2.1 – Операционный граф

3 Информационный граф

На рисунке 3.1 показан информационный граф.

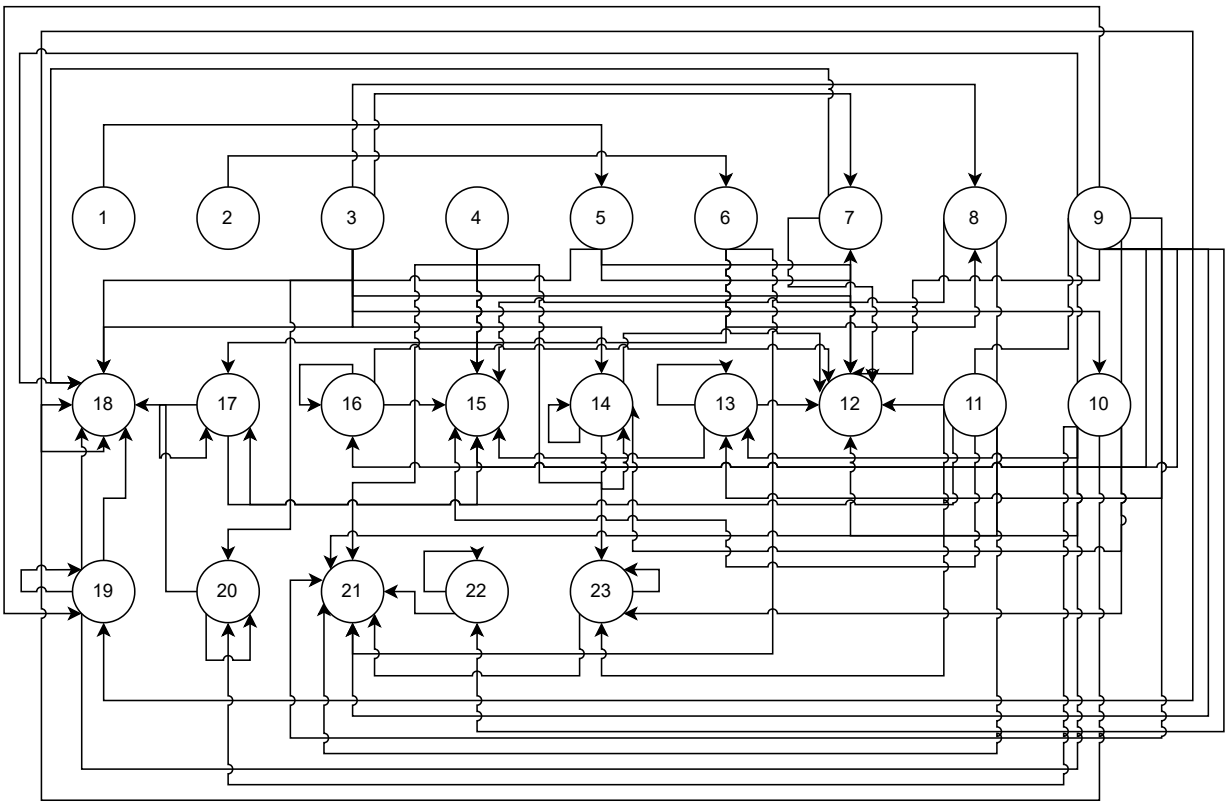

Рисунок 3.1 – Информационный граф

4 Граф операционной истории

На рисунке 4.1 показан граф операционной истории.

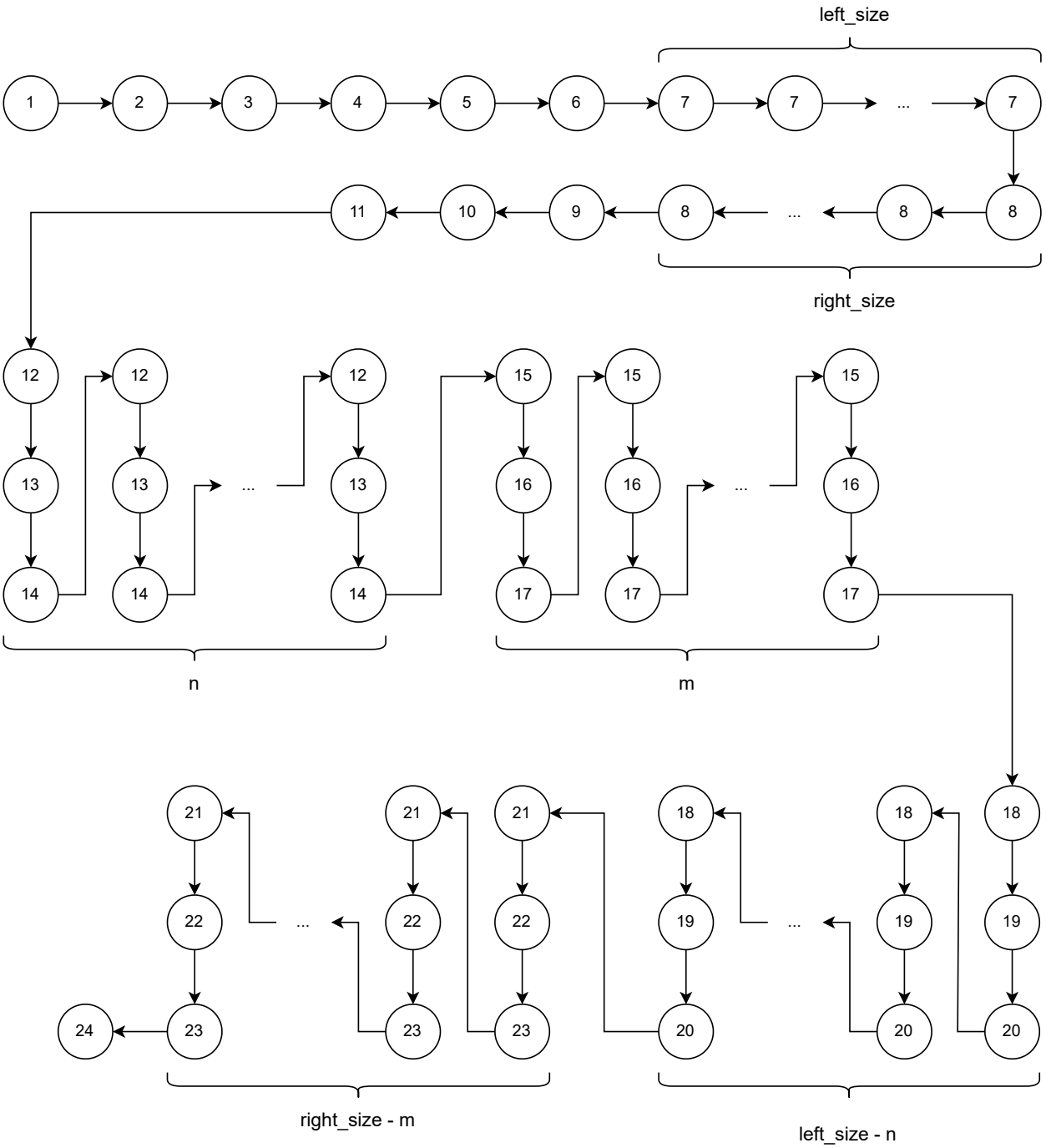

Рисунок 4.1 – Граф операционной истории

5 Граф информационной истории

На рисунке 5.1 показан граф информационной истории.

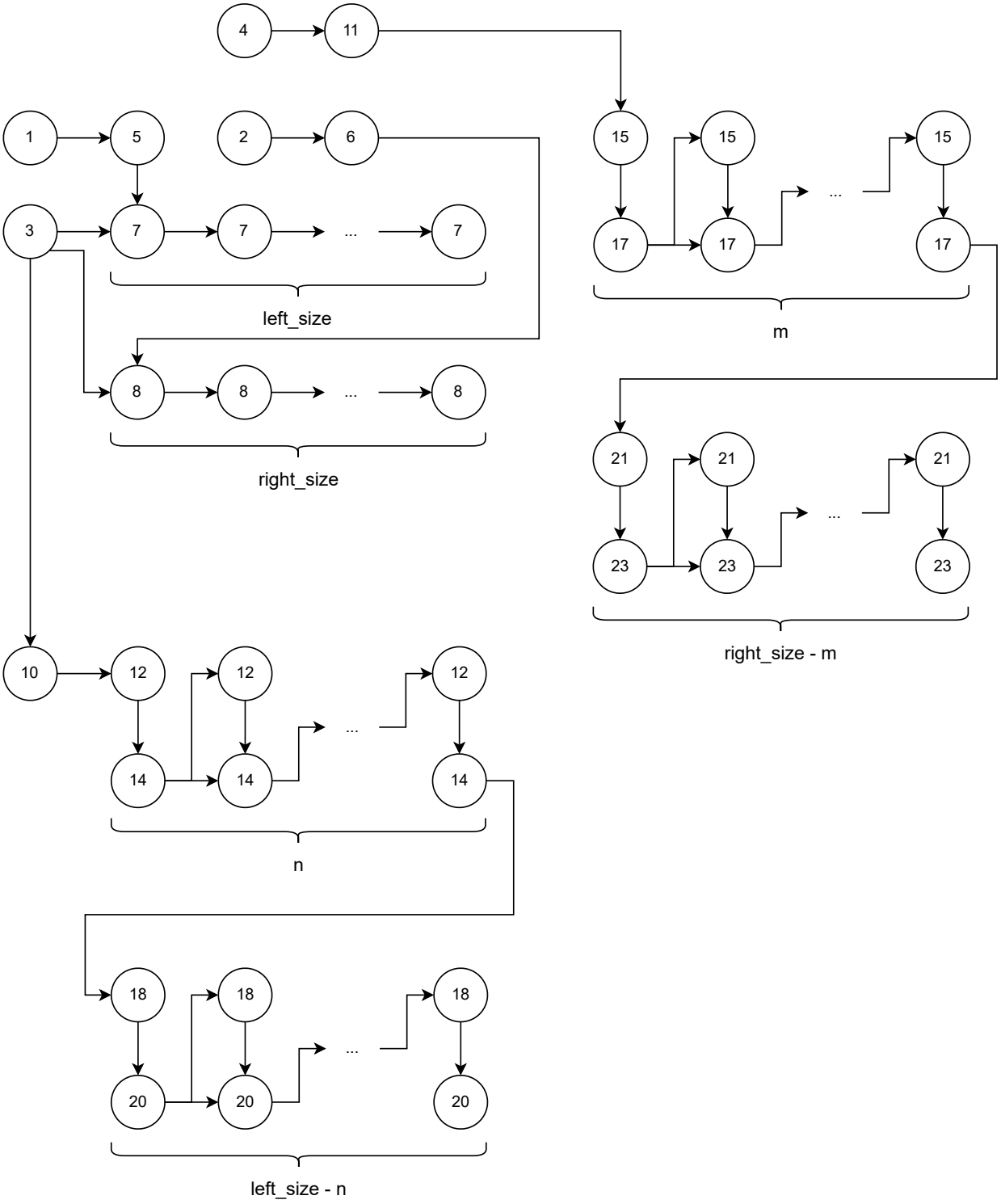

Рисунок 5.1 – Граф информационной истории