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

по курсу «Анализ Алгоритмов»

на тему: «Графовые модели алгоритмов»

Студент группы ИУ7-51Б _____ (Подпись, дата) Шубенина Д. В. (Фамилия И.О.)

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

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

Москва — 2023 г.

Содержание

1 Введение

1.1 Задание

Описать четырьмя графовыми моделями (ГУ, ИГ, ОИ, ИИ) последовательный алгоритм либо фрагмент алгоритма, содержащий от 15 значащих строк кода и от двух циклов, один из которых является вложенным в другой.

Вариант 21: реализуемый алгоритм — кластеризация нечетким алгоритмом c-средних.

1.2 Графовые модели программы

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

1) **Вершины**: процедуры, циклы, линейные участки, операторы, итерации циклов, срабатывание операторов и т. д.

2) **Дуги** отражают связь (отношение между вершинами).

Выделяют 2 типа отношений:

1) операционное отношение — по передаче управления;

2) информационное отношение — по передаче данных.

Граф управления:

1) **Вершины** — операторы.

2) **Дуги** — операционные отношения.

Информационный граф:

1) **Вершины** — операторы.

2) **Дуги** — информационные отношения.

Операционная история:

1) **Вершины** — срабатывание операторов.

2) **Дуги** — операционные отношения.

Информационная история:

1) **Вершины** — срабатывание операторов.

2) **Дуги** — информационные отношения.

2 Выполнение задания

2.1 Выбранный язык программирования

Для реализации алгоритма был выбран язык C++.

2.2 Реализация алгоритма

На листинге 2.1 представлен код реализуемого алгоритма.

Листинг 2.1 – Нечеткий алгоритм кластеризации с-средних

```
1     void c_means_clustering(std::vector<std::vector<double>>
        &cluster_centers, // -1
2     std::vector<std::vector<double>> &membership, // -2
3     const std::vector<std::vector<double>> &data, // -3
4     double m /* -4 */, double convergenceThreshold /* -5 */,
        double maxIterations /* -6 */)
5 {
6     int iterations = 0;

        // (1)
7     double delta = convergenceThreshold + 1.0;
                                                    // (2)
8     while (iterations < maxIterations && delta >
        convergenceThreshold)                     // (3)
9     {
10        for (size_t i = 0; i < data.size(); ++i)
                                                      // (4)
11        {
12            for (size_t j = 0; j < cluster_centers.size(); ++j)
                                          // (5)
13            {
14                double sum = 0.0;

                  // (6)
15                double dist1 = sqrt(pow(data[i][0] -
                    cluster_centers[j][0], 2) +
16                                    pow(data[i][1] -
                                       cluster_centers[j][1],
```

```
                                        2));          // (7)
17                  for (size_t k = 0; k < cluster_centers.size();
                        ++k)                          // (8)
18                  {
19                      double dist2 = sqrt(pow(data[i][0] -
                            cluster_centers[k][0], 2) +
20                                          pow(data[i][1] -
                                              cluster_centers[k][1],
                                              2));      // (9)
21                      sum += pow(dist1 / dist2, 2.0 / (m - 1.0));
                                                      // (10)
22                  }
23                  membership[i][j] = 1.0 / sum;

                        // (11)
24              }
25          }
26          for (size_t j = 0; j < cluster_centers.size(); ++j)
                                                      // (12)
27          {
28              double numeratorX = 0.0;

                    // (13)
29              double numeratorY = 0.0;

                    // (14)
30              double denominator = 0.0;

                    // (15)
31              for (size_t i = 0; i < data.size(); ++i)
                                                        // (16)
32              {
33                  double membershipPowM = pow(membership[i][j],
                        m);                           // (17)
34                  numeratorX += membershipPowM * data[i][0];
                                                        // (18)
35                  numeratorY += membershipPowM * data[i][1];
                                                        // (19)
36                  denominator += membershipPowM;
                                                                //
                        (20)
```

```
37              }
38              cluster_centers[j] = { numeratorX / denominator,
                    numeratorY / denominator }; // (21)
39          }
40          std::vector<std::vector<double>> old_cluster_centers =
                cluster_centers;
41          delta = 0.0;
42          for (size_t i = 0; i < cluster_centers.size(); ++i)
43          {
44              delta = sqrt(pow(data[i][0] -
                    old_cluster_centers[i][0], 2) +
45                             pow(data[i][1] -
                                   cluster_centers[i][1], 2));
46          }
47          ++iterations;
48      }
49  }
```

2.3 Графовые модели алгоритмов

2.3.1 Граф управления

На рисунке 2.1 показан граф управления программы.

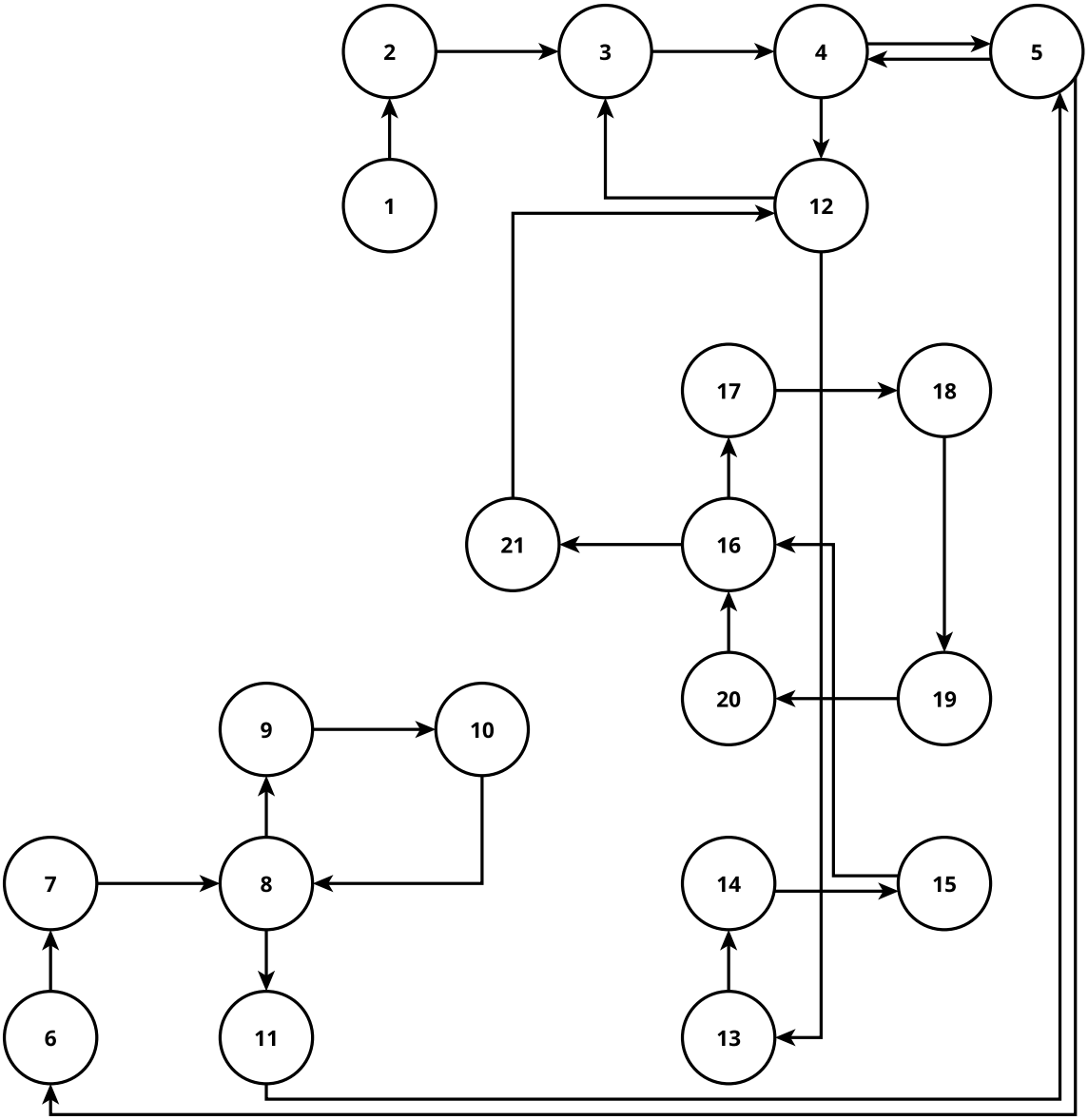

Рисунок 2.1 – Граф управления

2.3.2 Информационный граф

На рисунке 2.2 представлен информационный граф программы.

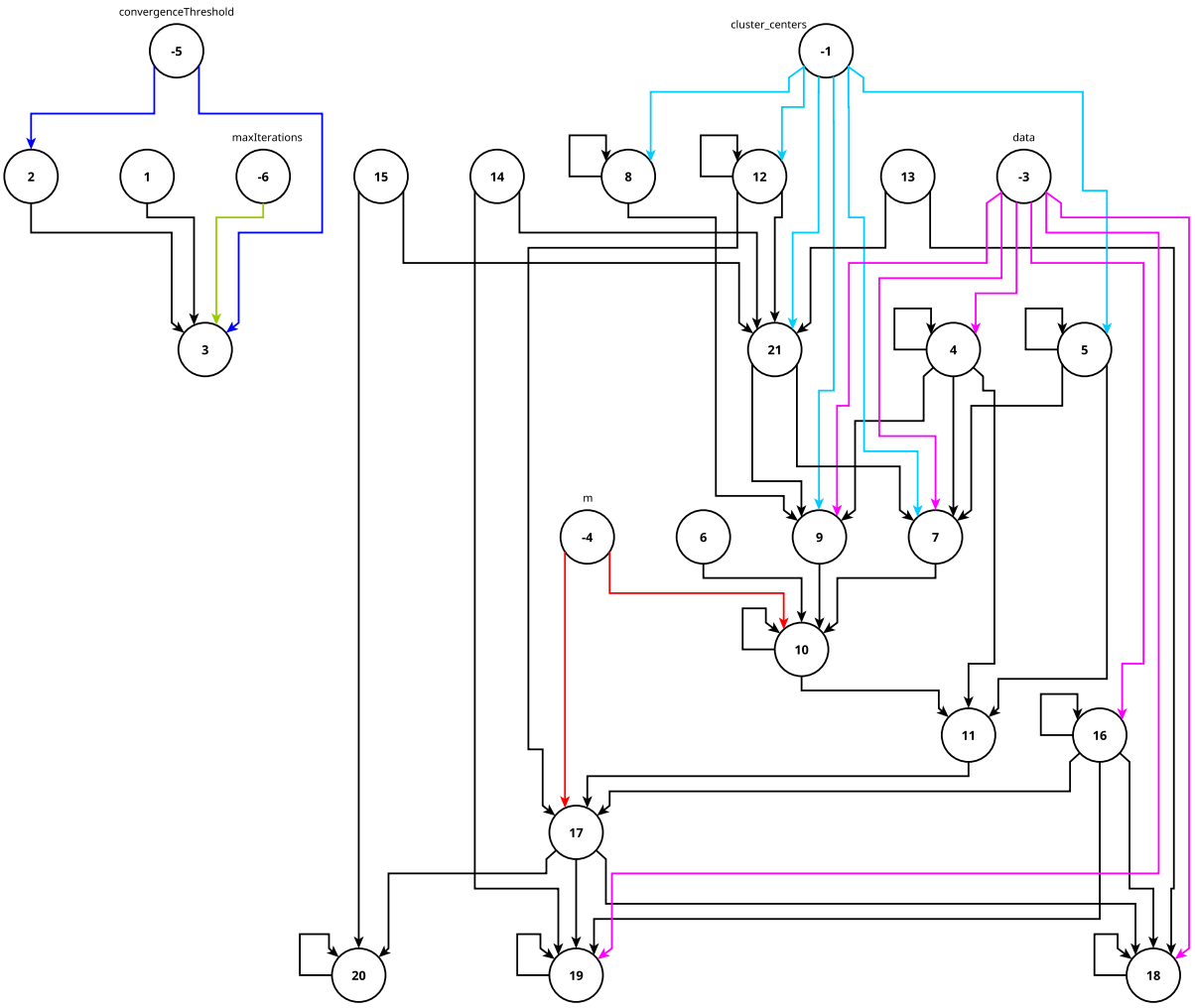

Рисунок 2.2 – Информационный граф

2.3.3 Операционная история

Операционная история программы представлена на рисунке 2.1 для следующих входных данных:

— `k = 1`;

— `m = 2`;

— `maxIterations = 2`;

— `convergenceThreshold = 0.1`;

— `data = [[5.1, 2.5], [1.4, 0.2]]`.

Для этих же данных будет составлена операционная история.

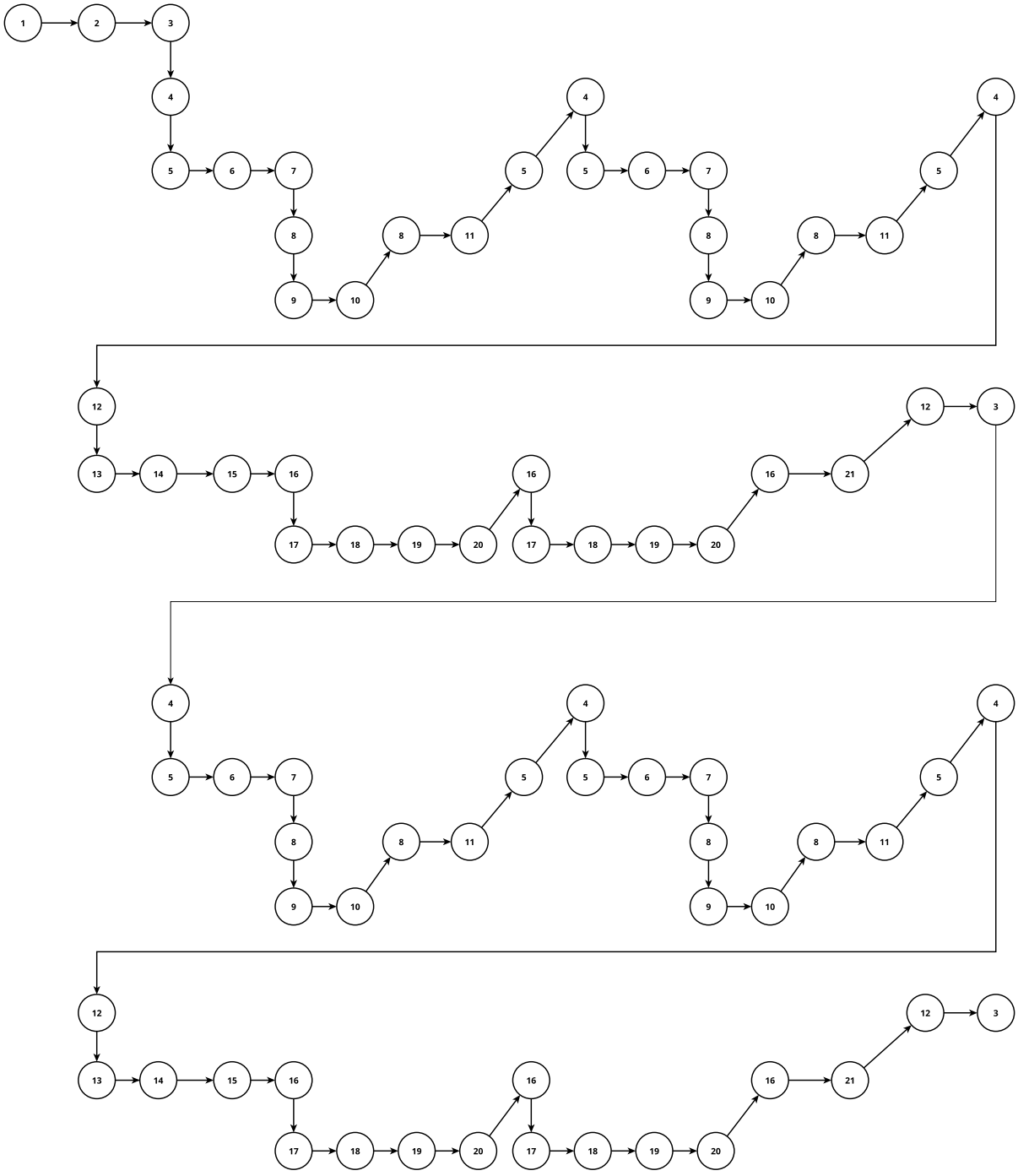

Рисунок 2.3 – Операционная история

2.3.4 Информационная история

На рисунке 2.4 представлена информационная история работы программы.

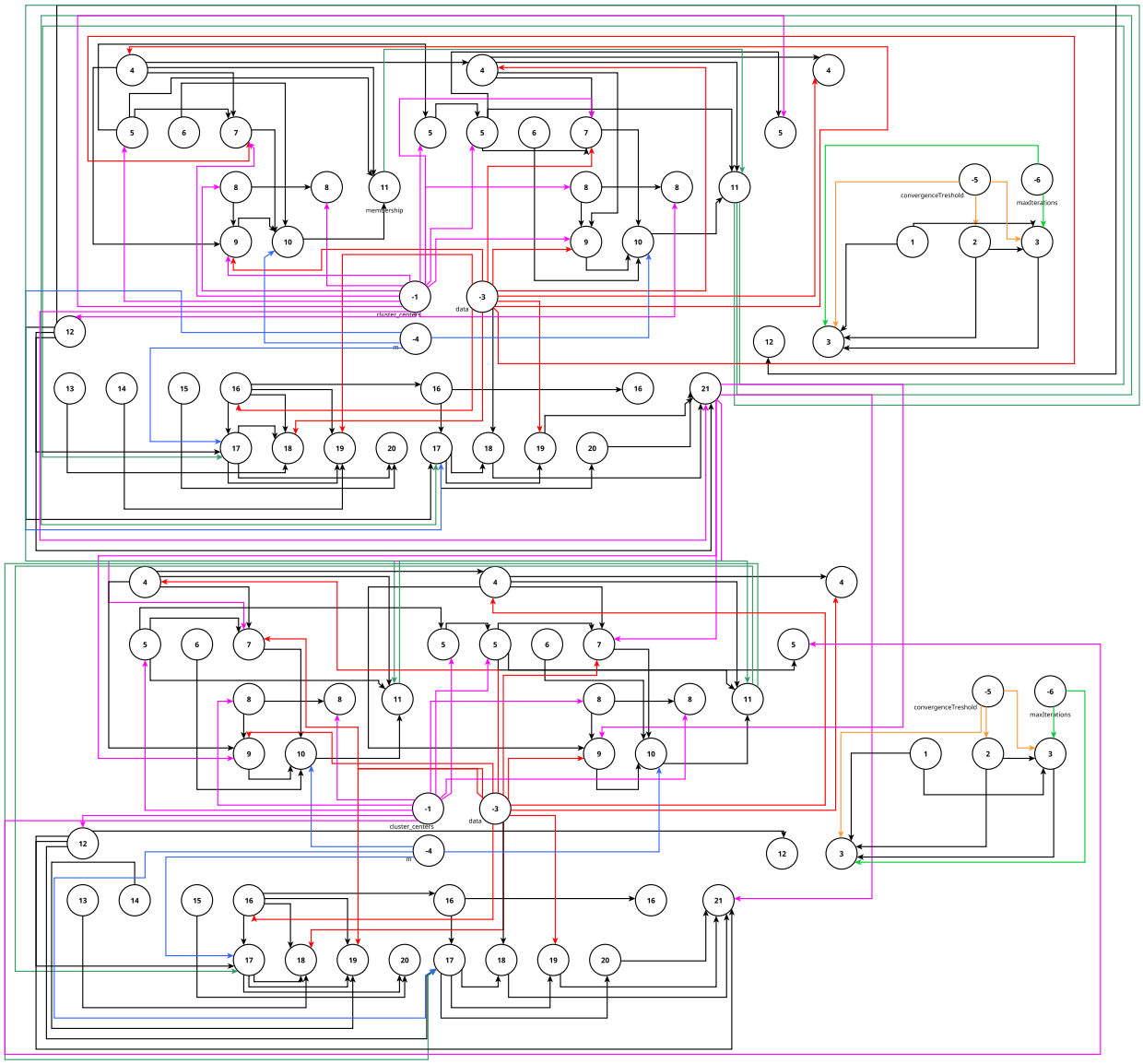

Рисунок 2.4 – Информационная история

Возможность распараллеливания

Алгоритм кластеризации с-средних может быть распараллелен следующими способами:

— параллельная инициализация центроидов;

— параллельное обновление центроидов после кластеризации.