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 ЛАБОРАТОРНОЙ РАБОТЕ №5
По курсу: "Анализ алгоритмов"

Студент Сукочева Алис
Группа ИУ7-53Б
Название предприятия МГТУ им. Н. Э. Баумана, каф. ИУ7
Тема Конвейер.

Студент:	_____	Сукочева А.
	подпись, дата	Фамилия, И.О.
Преподаватель:	_____	Волкова Л.Л.
	подпись, дата	Фамилия, И. О.
Преподаватель:	_____	Строганов Ю.В.
	подпись, дата	Фамилия, И. О.

Москва — 2020 г.

Оглавление

Введение **3**

1 Аналитическая часть **4**
1.1 Описание метода 4
1.2 Вывод 5

2 Конструкторская часть **6**
2.1 Разработка алгоритмов 6
2.2 Вывод 6

3 Технологическая часть **9**
3.1 Выбор ЯП 9
3.2 Сведения о модулях программы 9
3.3 Вывод 12

4 Экспериментальная часть **13**
4.1 Временные характеристики 13
4.2 Вывод 13

Заключение **16**

Список литературы **17**

Введение

Параллельные вычисления часто используются для увеличения скорости выполнения программ. Однако приемы, применяемые для однопоточных машин, для параллельных могут не подходить. Конвейерная обработка данных является популярным приемом при работе с параллельными машинами.

Целью данной работы является изучение и реализация метода конвейерных вычислений.

В рамках выполнения работы необходимо решить следующие задачи:

1. изучения основ конвейерной обработки данных;
2. применение изученных основ для реализации конвейерной обработки данных;
3. получения практических навыков;
4. получение статистики выполнения программы;
5. описание и обоснование полученных результатов;
6. выбор и обоснование языка программирования, для решения данной задачи.

1 | Аналитическая часть

Для данной лабораторной работы, которая предполагает конвейерную обработку, были выбраны три алгоритма для трех лент конвейера:

1. найти max в массиве;
2. найти min в массиве;
3. найти количество элементов в массиве, которые больше, чем (max - min) / 2.

1.1 Описание метода

Конвейер - устройство для непрерывного перемещения обрабатываемого изделия от одного рабочего к другому или для транспортировки грузов.

Конвейерное производство - система поточной организации производства на основе конвейера, при которой оно разделено на простейшие короткие операции, а перемещение деталей осуществляется автоматически.

В данной лабораторной работе мы выделяем три задачи, которые будут последовательно обрабатываться на конвейерной ленте. Каждая задача будет последовательно проходить три этапа обработки. Благодаря распараллеливанию мы сможем добиться того, что бы на всех трех этапах происходила обработка элемента, как в реальной жизни.

На рис 1.1 визуализирован конвейер.

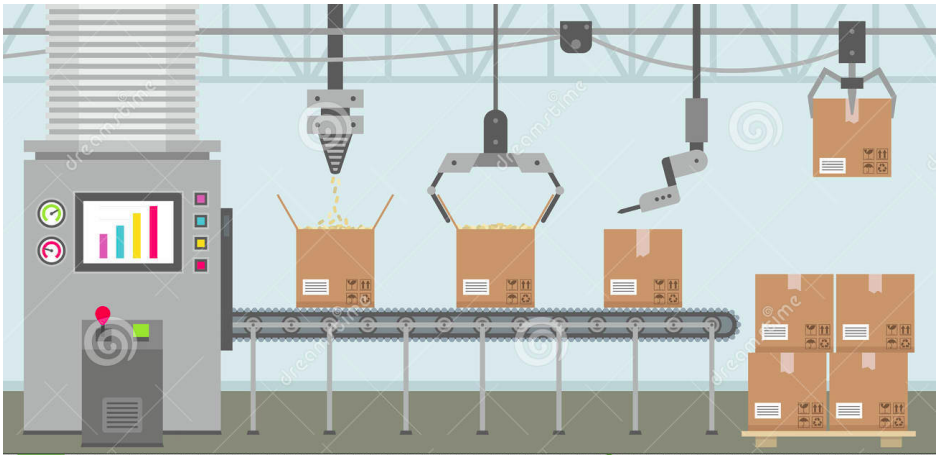

Рис. 1.1: Конвейер

1.2 Вывод

В данном разделе были рассмотрены основополагающие материалы, которые в дальнейшем потребуются при реализации конвейера.

2 | Конструкторская часть

2.1 Разработка алгоритмов

В данном разделе будут рассмотрены схемы.

На рис. 2.1 показана схема алгоритма главного потока.

На рис. 2.2 показана схема алгоритма конвейера.

2.2 Вывод

В данном разделе были рассмотрены схемы алгоритма главного потока 2.1 и схема алгоритма конвейера 2.2.

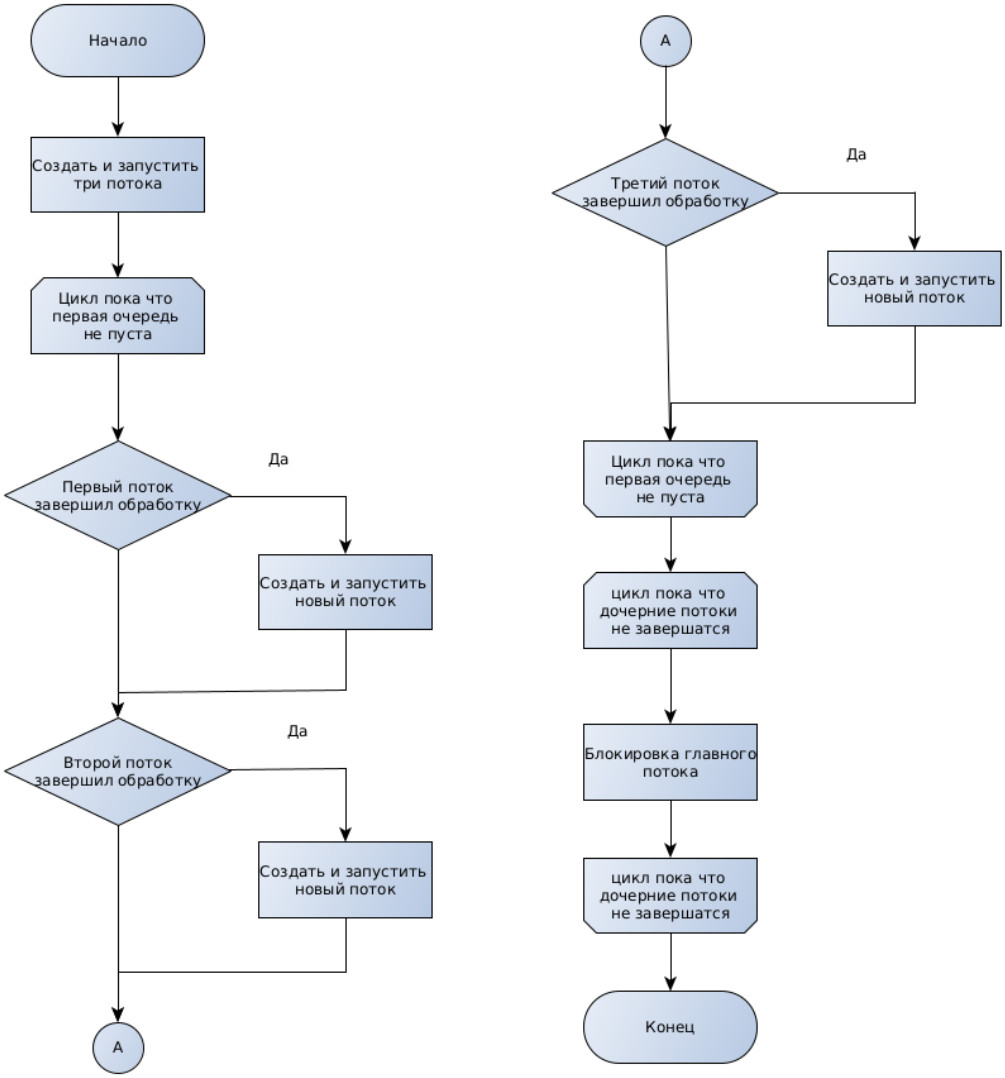

Рис. 2.1: Схема алгоритма главного потока

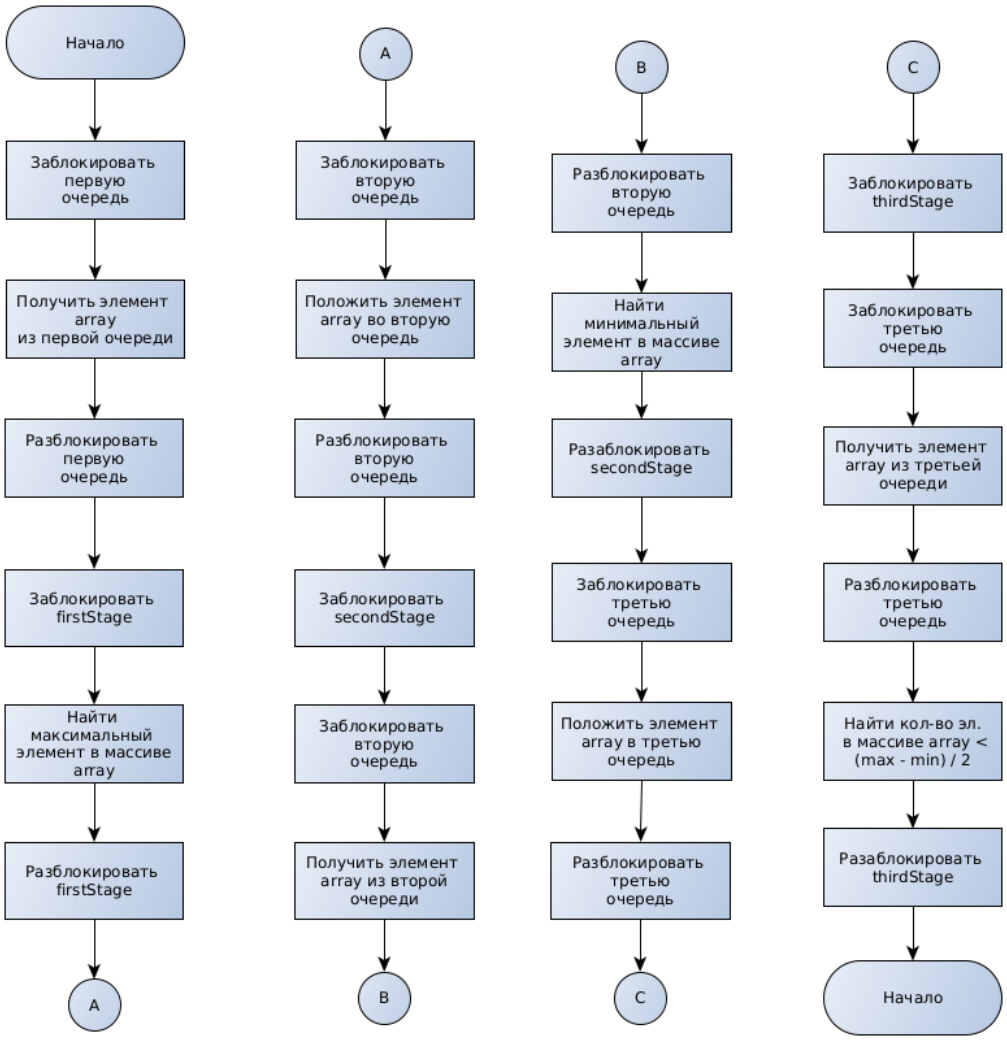

Рис. 2.2: Схема алгоритма конвейера

3 | Технологическая часть

3.1 Выбор ЯП

В данной лабораторной работе использовался язык программирования - C# [4]. Данный язык был выбран, потому что он является нативным. В качестве среды разработки я использовала Visual Studio Code [1]. Visual Studio Code подходит не только для Windows [2], но и для Linux [3], это причина, по которой я выбрала VS code, т.к. у меня установлена ОС Ubuntu 18.04.4 [5]. В моей архитектуре присутствует 8 ядер.

3.2 Сведения о модулях программы

Данная программа состоит из файла Program.cs, который содержит код программы.

На листингах 3.1-3.2 представлен основной код программы.

Листинг 3.1: Метод создания и запуска потоков

```
1 public static void MainTread(Queue<intPtr> queue)
2 {
3   Console.WriteLine("Process:\n");
4
5   ThreadArgs args = new ThreadArgs(queue);
6
7   Thread firstThread = new Thread(new
      ParameterizedThreadStart(Conveyor));
8   Thread secondThread = new Thread(new
      ParameterizedThreadStart(Conveyor));
9   Thread thirdThread = new Thread(new
      ParameterizedThreadStart(Conveyor));
```

```
10
11    firstThread.Start(args);
12    secondThread.Start(args);
13    thirdThread.Start(args);
14
15    while (args.firstQueue.Count != 0)
16    {
17      if (!firstThread.IsAlive)
18      {
19        firstThread = new Thread(new ParameterizedThreadStart
              (Conveyor));
20        firstThread.Start(args);
21      }
22
23      if (!secondThread.IsAlive)
24      {
25        secondThread = new Thread(new
              ParameterizedThreadStart(Conveyor));
26        secondThread.Start(args);
27      }
28
29      if (!thirdThread.IsAlive)
30      {
31        thirdThread = new Thread(new ParameterizedThreadStart
              (Conveyor));
32        thirdThread.Start(args);
33      }
34    }
35
36    firstThread.Join(); secondThread.Join(); thirdThread.Join
        ();
37  }
```

Листинг 3.2: Конвейер

```
1  public static void Conveyor(object obj)
2  {
3    ThreadArgs args = (ThreadArgs)obj;
4    int max, min, count;
5    intPtr array;
6
```

```
7   lock (args.firstQueue)
8   {
9     // Get array from the first queue.
10    array = args.firstQueue.Dequeue();
11  }
12
13  lock (firstStage)
14  {
15    // The first tape is running.
16    max = FindMax(array);
17  }
18
19  lock (args.secondQueue)
20  {
21    // Added element to the second queue.
22    args.secondQueue.Enqueue(array);
23  }
24
25  lock (secondStage)
26  {
27    // The second tape is running.
28    lock (args.secondQueue)
29    {
30      // Get array from the second queue.
31      array = args.secondQueue.Dequeue();
32    }
33    min = FindMin(array);
34  }
35
36  lock (args.thirdQueue)
37  {
38    args.thirdQueue.Enqueue(array);
39  }
40
41  lock (thirdStage)
42  {
43    lock (args.thirdQueue)
44    {
45      array = args.thirdQueue.Dequeue();
46    }
```

```
47      count = FindCount(array, (max - min) / 2);
48    }
49  }
```

3.3 Вывод

В данном разделе были разобраны листинги рис 3.1-3.2, показывающие работу конвейера.

4 | Экспериментальная часть

В данном разделе будет произведено сравнение последовательной реализации трех алгоритмов и конвейера с использованием многопоточности. А также показана статистика.

4.1 Временные характеристики

Для сравнения возьмем 10 массивов размерностью [5, 10, 25, 50, 100, 250, 1000]. Воспользуемся усреднением массового эксперимента.

Результат сравнения последовательной реализации трех алгоритмов и конвейера, с использованием многопоточности, представлен на рис. 4.1.

По результатам эксперимента видно, что время испольнения конвейерной реализации значительно меньше, чем исполнение традиционной реализации. В начале традиционная реализация выигрывает по времени, потому что при маленьких задачах тратится много времени на создание потоков и ожидание доступа к переменной, что значительно снижает работоспособность программы.

На рис. 4.2 приведена статистика.

4.2 Вывод

В данном разделе было произведено сравнение последовательной реализации трех алгоритмов и конвейера с использованием многопоточности. По результатам исследования конвейерную обработку нет смысла применять на задачах, занимающих мало времени. Статистика показала, что конвейерная обработка работает правильно.

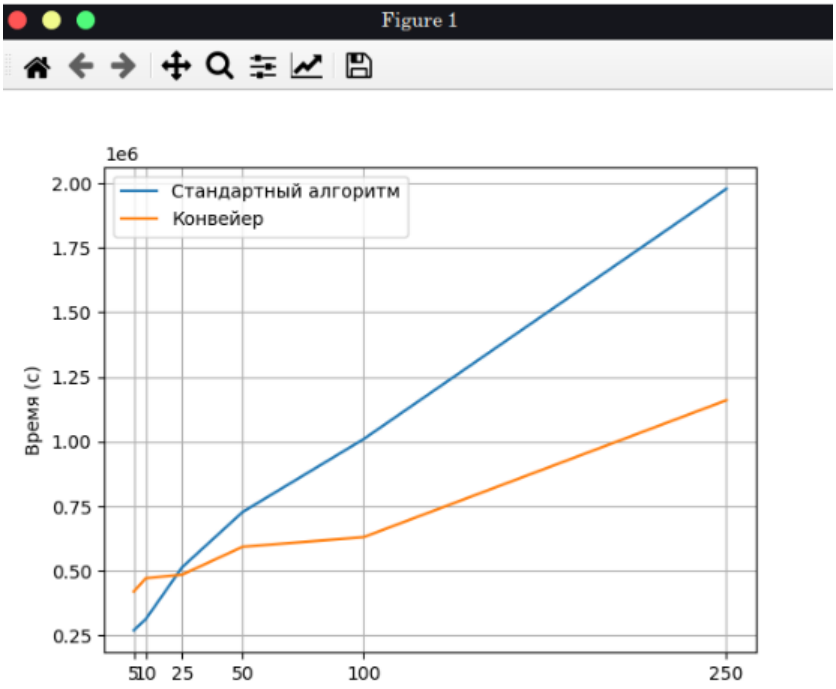

Рис. 4.1: Временные характеристики

```
Лента 1 начало Поток 1    637413148520705300
Лента 1 конец  Поток 1    637413148520798420
Лента 2 начало Поток 1    637413148520800150
Лента 1 начало Поток 3    637413148520800470
Лента 1 конец  Поток 3    637413148520854820
Лента 1 начало Поток 2    637413148520855610
Лента 2 конец  Поток 1    637413148520857220
Лента 3 начало Поток 1    637413148520858050
Лента 2 начало Поток 3    637413148520858340
Лента 2 конец  Поток 3    637413148520908820
Лента 3 конец  Поток 1    637413148520920160
Лента 3 начало Поток 3    637413148520921030
Лента 1 конец  Поток 2    637413148520945990
Лента 2 начало Поток 2    637413148520946480
Лента 1 начало Поток 1    637413148520947120
Лента 3 конец  Поток 3    637413148520969300
Лента 1 конец  Поток 1    637413148521000900
Лента 1 начало Поток 3    637413148521001420
Лента 2 конец  Поток 2    637413148521006570
Лента 3 начало Поток 2    637413148521006970
Лента 2 начало Поток 1    637413148521007080
Лента 1 конец  Поток 3    637413148521073730
Лента 3 конец  Поток 2    637413148521093490
Лента 2 конец  Поток 1    637413148521096710
Лента 3 начало Поток 1    637413148521097020
Лента 2 начало Поток 3    637413148521097570
Лента 3 конец  Поток 1    637413148521146140
Лента 2 конец  Поток 3    637413148521147650
Лента 3 начало Поток 3    637413148521147860
Лента 3 конец  Поток 3    637413148521195840
```

Рис. 4.2: Временные характеристики

Заключение

В данной лабораторной работе были рассмотрены основополагающие материалы которые в дальнейшем потребовались при реализации конвейера. Были рассмотрены схемы (рис. 2.1 - 2.2) показывающие алгоритм конвейера. Также были разобраны листинги рис 3.1-3.2, показывающие работу контейнера и был приведен рис. 4.2, показывающий корректную работу конвейера. Был произведен сравнительный анализ рис. 4.1.

В рамках выполнения работы решены следующие задачи:

1. изучили освоили конвейерную обработку данных;
2. применили изученные основы для реализации конвейерной обработки данных;
3. получили практические навыки;
4. получили статистику выполнения программы;
5. описали и обосновали полученные результаты;
6. выбрали и обосновали языка программирования, для решения данной задачи.

Литература

[1] Visual Studio Code [Электронный ресурс], режим доступа: https://code.visualstudio.com/ (дата обращения: 02.10.2020)

[2] Windows [Электронный ресурс], режим доступа:https://www.microsoft.com/ru-ru/windows (дата обращения: 02.10.2020)

[3] Linux [Электронный ресурс], режим доступа:https://www.linux.org.ru/ (дата обращения: 02.10.2020)

[4] Руководство по языку C#[Электронный ресурс], - режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/

[5] Ubuntu 18.04 [Электронный ресурс], режим доступа:https://releases.ubuntu.com/18.04/ (дата обращения: 02.10.2020)