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»

Отчёт

по лабораторной работе № 5

Название: Многопоточная реализация конвейера

Дисциплина: Анализ алгоритмов

Студент ИУ7-55Б _____ Д.В. Сусликов

(Группа) (Подпись, дата) (И.О. Фамилия)

Преподователь _____ Л.Л. Волкова

(Подпись, дата) (И.О. Фамилия)

Москва, 2020

Содержание

Введение

Цель работы: создание системы конвейерных вычислений.

В ходе лабораторной работы требуется:

1) дать описание алгоритма реализации конвейерных вычислений;
2) реализовать данный алгоритм;
3) провести его тестирование.

1 Аналитический раздел

В данном разделе представлены принципы конвейерных вычислений и параллельных.

1.1 Общие сведения

Конвейерное производство — система поточной организации производства на основе конвейера, при которой оно разделено на простейшие короткие операции, а перемещение деталей осуществляется автоматически. Это такая организация выполнения операций над объектами, при которой весь процесс воздействия разделяется на последовательность стадий с целью повышения производительности путём одновременного независимого выполнения операций над несколькими объектами, проходящими различные стадии.

Конвейером также называют средство продвижения объектов между стадиями при такой организации[4].

1.2 Параллельные вычисления

Параллельные вычисления – способ организации компьютерных вычислений, при котором программы разрабатываются как набор взаимодействующих вычислительных процессов, работающих параллельно (одновременно). Термин охватывает совокупность вопросов параллелизма в программировании, а также создание эффективно действующих аппаратных реализаций. Теория параллельных вычислений составляет раздел прикладной теории алгоритмов[5].

Вывод

По итогу, были разобраны общая информация о конвейерном производстве и конвейерах и суть параллельных вычислений.

2 Конструкторский раздел

В данном разделе представлена схема алгоритмов обработки элементов линии конвейера, а также описана сама система.

2.1 Разработка алгоритма

На Рисунке 1 изображена схема алгоритмов обработки элементов линии конвейера.

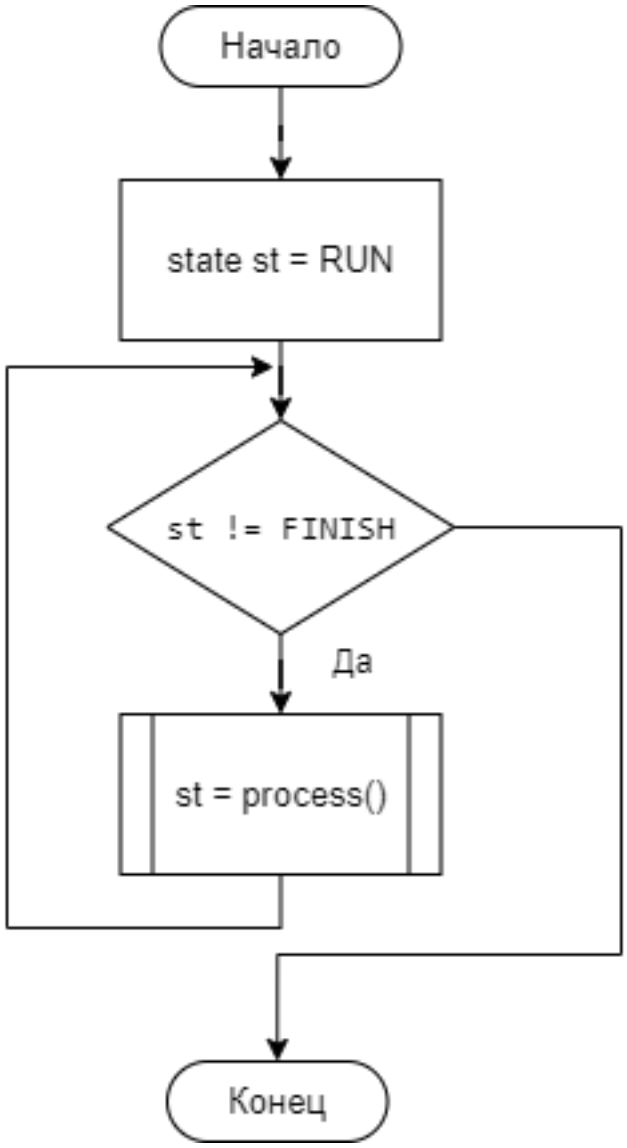

Рисунок 1 – Схема алгоритма старта и процесса линии конвейера

На Рисунке 2 можно увидеть схему алгоритма обработки элементов линии конвейера.

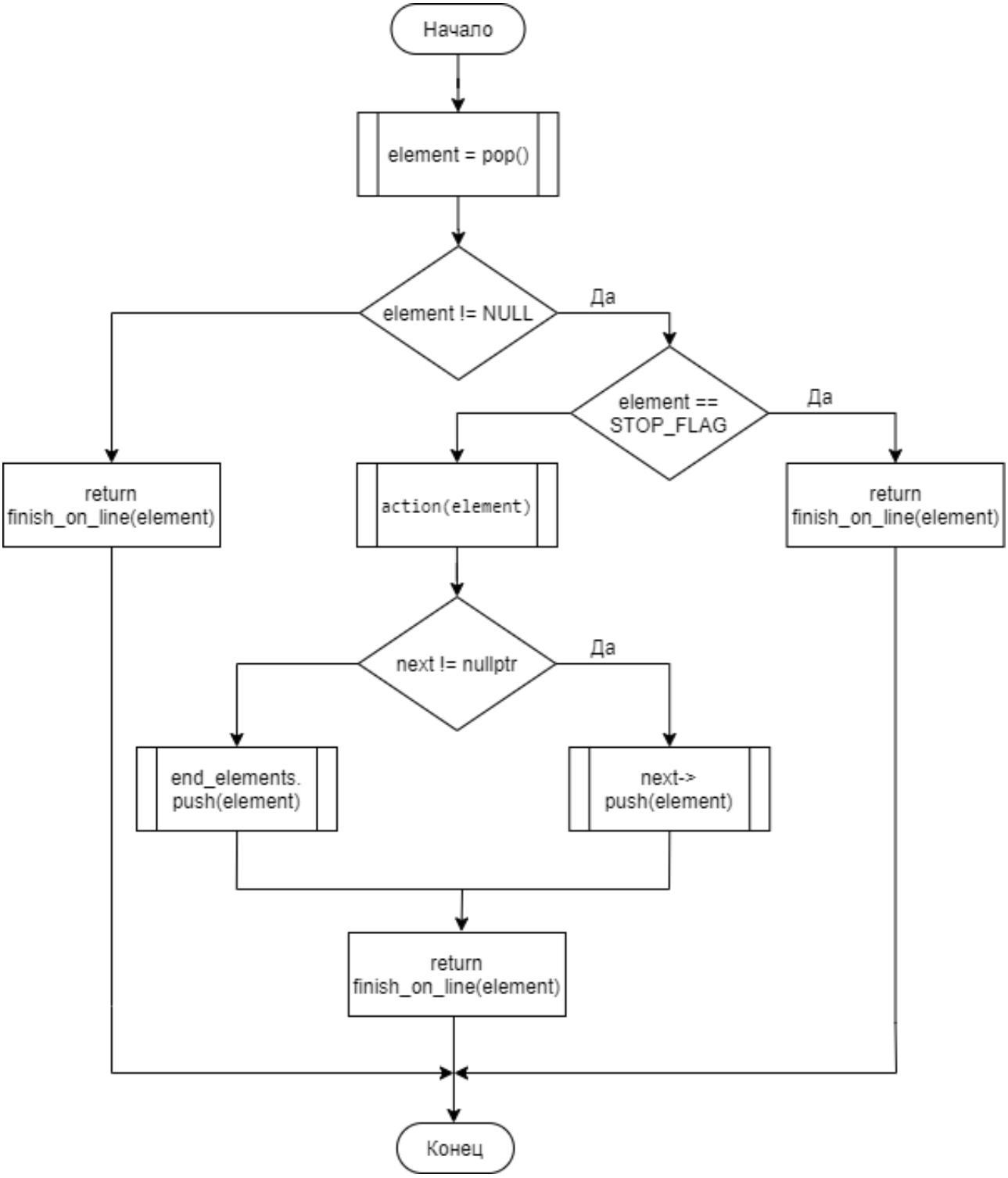

Рисунок 2 – Схема алгоритма обработки элементов линии конвейера

2.2 Описание системы

Система состоит из 3 параллельно работающих линий. Каждая линия имеет указатель на следующую, кроме последней, которая указывает на $nullptr$. Изначально задаётся очередь элементов, последний из которых является флагом окончания очереди $STOP\_FLAG$. Каждый элемент поочередно записывается на первую линию, обрабатывается и посылается на следующую линию. Время о поступлении и выхода с линии и другая информация своевременно выводятся на экран. С последней линии элементы записываются в результирующий массив $end\_elements$ для последующего вывода.

Вывод

Таким образом, были разобраны схема алгоритмов обработки элементов линии конвейера, а также сама система.

3 Технологический раздел

В данном разделе даны общие требования к программе, средства реализации и реализация алгоритмов.

3.1 Общие требования

Требование:

1) аргументы должны последовательно проходить линии в заданном порядке;
2) каждая линия должна работать в своем потоке;
3) последним элементом должен быть флаг, при поступлении которого, линия должна завершить свою работу;
4) при пустой очереди линии, она должна ожидать поступления нового элемента;

5) в результате из последней линии должен вернуться массив аргументов, у которого порядок совпадает с начальным.

3.2 Средства реализации

В лабораторной работе был использован язык C++[1], так как он известен, и на нём было написано множество предыдущих работ.

Среда разработки - Qt[2].

Для замеров процессорного времени была использована функция $clock()$[3].

3.3 Реализация алгоритмов

В Листинге 1 реализован алгоритм старта линии конвейера.

Листинг 1 - Алгоритм старта линии конвейера

```
1  void ConveyerLine::start_line()
2  {
3    state st = RUN;
4    while (st != FINISH)
5      st = process();
6  }
```

В Листинге 2 реализован алгоритм обработки аргумента линии.

Листинг 2 - Алгоритм обработки аргумента линии

```
1  state ConveyerLine::process()
2  {
3    int element = pop();
4    if (element != NULL)
5    {
6      if (element == STOP_FLAG)
7        return finish_on_line(element);
8
9      action(element);
10
11     if (next != nullptr)
12       next->push(element);
13     else
14       end_elements.push(element);
15   }
16   else
17     return STOP;
18
19   return RUN;
20 }
```

В Листинге 3 реализовано добавление элемента в очередь.

Листинг 3 - Добавление элемента в очередь

```
1  void ConveyerLine::push(int element)
2  {
3    mute.lock();
4    elements.push(element);
5    mute.unlock();
6  }
```

В Листинге 4 показано взятие элемента из очереди.

Листинг 4 - Взятие элемента из очереди.

```
1  int ConveyerLine::pop()
2  {
3    int element = NULL;
4    mute.lock();
5    int size = elements.size();
6    if (size > 0)
7    {
8      element = elements.front();
9      elements.pop();
10   }
11   mute.unlock();
12   return element;
13 }
```

В Листинге 5 реализована обработка объекта и выводы замеров времени.

Листинг 5 - Обработка объекта и выводы замеров времени

```
1  void ConveyerLine::action(int element)
2  {
3    cout << "Line " << line_num << " | Element " << element << "
       ON line at " << clock() << endl;
4    this_thread::sleep_for(chrono::milliseconds(action_time));
5    cout << "Line " << line_num << " | Element " << element << "
       OUT of line at " << clock() << endl;
6  }
```

В Листинге 6 показано завершение работы линии.

Листинг 6 - Завершение работы линии

```
state ConveyerLine::finish_on_line(int element)
{
  if (next != nullptr)
    next->push(element);

  return FINISH;
}
```

В Листинге 7 реализован вывод полностью обработанных элементов.

Листинг 7 - Вывод полностью обработанных элементов

```
void ConveyerLine::get_ended_elements()
{
  if (next == nullptr)
  {
    int len = end_elements.size();
    for (int _ = 0; _ < len; _++)
    {
      cout << end_elements.front() << " ";
      end_elements.pop();
    }
  }
  else
    cout << "Not last line!";
}
```

Вывод

Таким образом, были разобраны требования к программе, описаны средства реализации, и приведен код операций связанных с работой конвейера.

4 Экспериментальный раздел

В данном разделе представлен результаты работы программы и показано параллельное выполнение операций.

4.1 Пример работы программы

На Рисунке 3 показан результат работы программы.

```
Line 1 | Element 1 ON line at 2
Line 1 | Element 1 OUT of line at 1003
Line 1 | Element 2 ON line at 1004
Line 2 | Element 1 ON line at 1005
Line 1 | Element 2 OUT of line at 2005
Line 1 | Element 3 ON line at 2006
Line 2 | Element 1 OUT of line at 3006
Line 1 | Element 3 OUT of line at 3006
Line 1 | Element 4 ON line at 3007
Line 3 | Element 1 ON line at 3008
Line 2 | Element 2 ON line at 3008
Line 1 | Element 4 OUT of line at 4007
Line 1 | Element 5 ON line at 4007
Line 1 | Element 5 OUT of line at 5008
Line 2 | Element 2 OUT of line at 5009
Line 2 | Element 3 ON line at 5009
Line 3 | Element 1 OUT of line at 6009
Line 3 | Element 2 ON line at 6010
Line 2 | Element 3 OUT of line at 7010
Line 2 | Element 4 ON line at 7010
Line 3 | Element 2 OUT of line at 9011
Line 3 | Element 3 ON line at 9011
Line 2 | Element 4 OUT of line at 9012
Line 2 | Element 5 ON line at 9012
Line 2 | Element 5 OUT of line at 11012
Line 3 | Element 3 OUT of line at 12013
Line 3 | Element 4 ON line at 12014
Line 3 | Element 4 OUT of line at 15014
Line 3 | Element 5 ON line at 15015
Line 3 | Element 5 OUT of line at 18016
RESULT OBJECTS
1 2 3 4 5
```

Рисунок 3 – Результат работы программы

Вывод

По результаты работы программы можно сделать вывод, что действия, а именно начало обработки элементов и конец, выполняются параллельно. Конечный результат показывает, что последовательность элементов сохраняется.

Заключение

В ходе выполнения лабораторной работы были изучены возможности параллельных вычислений и применены на примере конвейерной системы. Была разработан и описан конвейер с параллельно работающими линиями. Были даны ответствующие схемы.

Цель работы достигнута, все поставленные задачи выполнены.

Литература

1) Бьерн Страуструп. Язык программирования C++. -URL:

   https://codernet.ru/books/c_plus/bern_straustrup_yazyk_programmirovaniya_c_specialnoe_izdanie/

   (дата обращения: 01.10.2020). Текст: электронный.

2) Qt. -URL:

   https://www.qt.io/ (дата обращения: 01.10.2020). Текст: электронный.

3) Функция $clock$. -URL:

   https://docs.microsoft.com/ru-ru/cpp/c-runtime-library/reference/clock?view=vs-2019 (дата обращения: 01.10.2020). Текст: электронный.

4) Конвейерное производство. -URL:

   https://kartaslov.ru/карта-знаний/Конвейерное+производство
   (дата обращения: 23.11.2020). Текст: электронный.

5) Параллельные вычисления -URL:

   https://ru.bmstu.wiki/ (дата обращения: 13.11.2020). Текст: электронный.