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 №6
по курсу «Анализ алгоритмов»

«Муравьиный алгоритм»

Студент Маслова Марина Дмитриевна

Группа ИУ7-53Б

Оценка (баллы) _____

Преподаватель Волкова Лилия Леонидовна

2021 г.

Содержание

Введение

Целью данной работы является

Для достижения поставленной цели необходимо выполнить следующие **задачи**:

- описать конвейерную обработку данных;
- описать алгоритмы, реализуемые на каждом этапе конвейера;
- описать функциональные требования;
- разработать описанные алгоритмы;
- реализовать алгоритмы всех этапов;
- реализовать конвейерную обработку данных;
- реализовать линейную обработку данных;
- провести тестирование реализованных алгоритмов;
- провести сравнительный анализ алгоритмов по времени работы реализаций;
- сделать выводы по полученным результатам.

1 Аналитическая часть

В данном разделе представлено теоретическое описание конвейерной обработки данных и алгоритмов каждой стадии обработки.

1.1 Конвейерная обработка данных

1.2 Вывод

В данном разделе была описана модель конвейерной обработки данных, также были представлены описания алгоритмов, выполняющихся на каждой ленте конвейера. В качестве реализуемого поэтапного алгоритма была выбрана генерация зашифрованных сообщений. Из представленных описаний можно предъявить ряд требований к разрабатываемому программному обеспечению:

- на вход должно подаваться количество обрабатываемых заявок;
- при неверных входных данных должно выдаваться сообщение об ошибке;
- на выходе должны выдаваться логи работы каждого из методов обработки (последовательного и конвейерного).

2 Конструкторская часть

В данном разделе разрабатываются алгоритмы генерации сообщений, перестановки букв в словах сообщений и шифра Веженера, алгоритмы линейной обработки и каждого потока конвейерной обрабокти, включая главный, также описывается структура программы и способы её тестироваия.

2.1 Разработка алгоритмов

На рисунке **??** представлена схема алгоритма генерации сообщений, на рисунках **??-??** – схема алгоритма перестановки букв в словах, на рисунке **??** – схема алгоритма применения шифра Веженера.

На рисунке **??** представлена схема алгоритма линейной обработки заявок. На рисунке **??** представлена схема главного потока, запускающего потоки этапов конвейера, схемы алгоритмов которых представлены на рисунках **??-??**.

2.2 Структура разрабатываемого ПО

Для реализации разрабатываемого программного обеспечения будет использоваться метод структурного программирования. Каждый из алгоритмов будет представлен отдельной функцией, при необходимости будут выделены подпрограммы для каждой из них. Также будут реализованы функции для ввода-вывода и функция, вызывающая все подпрограммы для связности и полноценности программы.

2.3 Классы эквивалентности при тестировании

Для тестирования программного обеспечения во множестве тестов будут выделены следующие классы эквивалентности:

- отрицательное число;
- ноль заявок;
- нечисловые данные;
- разные порядки верного количества заявок.

2.4 Вывод

В данном разделе были разработаны алгоритмы этапов обработки, а также алгоритмы линейной и конвейерной обработки заявок, была описана структура разрабатываемого ПО. Для дальнейшей проверки правильности работы программы были выделены классы эквивалентности тестов.

3 Технологическая часть

В данном разделе описаны требования к программному обеспечению, средства реализации, приведены листинги кода и данные, на которых будет проводиться тестирование.

3.1 Требования к программному обеспечению

Программа должна предоставлять следующие возможности:

- ввода количества обрабатываемых заявок;
- вывода лога работы программы для введенного количества заявок, включающих для каждой заявки время начала и конца обработки на каждой из лент;
- получения времени обработки всех заявок.

3.2 Средства реализации

3.3 Листинги кода

В данном подразделе представлены листинги кода алгоритмов:

– реализация генерации сообщений (листинг **??**);

– реализация перестановки букв в каждом слове в обратном порядке (листинг **??**);

– реализация шифра Веженера (листинг **??**);

– последовательная реализация конвейера (листинг **??**);

– параллельная реализация конвейера (листинг **??**).

3.4 Описание тестирования

В таблице 3.1 приведены функциональные тесты программы.

Таблица 3.1 – Функциональные тесты

Количество заявок	**Ожидаемый результат**
-1	Сообщение об ошибке
0	Сообщение об ошибке
j	Сообщение об ошибке
10	Лог работы каждого из алгоритмов
100	Лог работы каждого из алгоритмов
1000	Лог работы каждого из алгоритмов

3.5 Вывод

В данном разделе были реализованы последовательный и конвейрный алгоритмы поэтапной генерации зашифрованных сообщений. Также были написаны тесты для каждого класса эквивалентности, описанного в конструкторском разделе.

4 Исследовательская часть

4.1 Технические характеристики

Технические характеристики устройства, на котором выполнялось тестирование:

– Операционная система: Manjaro [1] Linux x86_64.

– Память: 8 GiB.

– Процессор: Intel® Core™ i5-8265U, 4 физических ядра, 8 логических ядра[2].

Тестирование проводилось на ноутбуке, включенном в сеть электропитания. Во время тестирования ноутбук был нагружен только встроенными приложениями окружения, окружением, а также непосредственно системой тестирования.

4.2 Примеры работы программы

На рисунке **??** представлены результаты работы последовательной обработки, на рисунке **??** – конвейерной.

4.3 Результаты тестирования

Программа была протестирована на входных данных, приведенных в таблице 3.1. Полученные результаты работы программы совпали с ожидаемыми результатами.

4.4 Постановка эксперимента по замеру времени

Для оценки времени работы последовательной и конвейерной реализации алгоритмов шифрования был проведен эксперимент, в котором определялось влияние количества заявок и количества слов в сообщении на время работы каждого из алгоритмов. Тестирование проводилось на количестве заявок от 5 до 10 с шагом 5, от 25 до 100 с шагом 25 и от 100 до 1000 с

шагом 250, а количество слов принимало значения $5, 10, 25, 50, 75, 100$. Время работы на каждом из значений было получено с помощью *бенчмарков*[3], являющимися встроенными средствами языка Go. В них количество повторов измерений времени изменяется динамически до тех пор, пока не будет получен стабильный результат.

Результаты эксперимента были представлены в виде таблиц и графиков, приведенных в следующем подразделе.

4.5 Результаты эксперимента

В таблице **??** представлены результаты измерения времени работы последовательной и конвейерной реализаций в зависимости от числа заявок. На рисунке **??** представлен соответствующий график.

В таблице **??** представлены результаты измерения времени работы последовательной и конвейерной реализаций в зависимости от числа слов в сообщениях при фиксированном числе заявок, равном 50. На рисунке **??** представлен соответствующий график.

4.6 Вывод

По результатам эксперимента можно сделать следующие выводы:

- при количестве заявок до 10 последовательная и конвейерная реализация генерации зашифрованных сообщений отрабатывают за одинаковое время, а при количестве заявок до 5 последовательная реализация работает 1.3 раза быстрее, что объясняется затратами на передачу данных между лентами с помощью очередей/каналов в конвейерной реализации;
- при большем количестве заявок от 25 конвейерная реализация работает в 1.7 раза быстрее последовательной;
- при фиксированном количестве заявок при различных количествах слов в сообщениях конвейерная реализация работает в 1.7 раза быстрее последовательной.

Таким образом, для обработки количества заявок большего 10 для достижения оптимальной скорости вычислений необходимо использовать конвейерную реализацию. Если работа просходит с количеством заявок до 10 достаточно последовательной реализации, то есть нет необходимости реализовывать более сложный конвейерный алгоритм.

Заключение

В ходе исследования был проведен сравнительный аналализ последовательной и конвейерной реализации алгорима поэтапного шифрования сообщений. В результате исследования было выяснено, что при количестве заявок более 10 конвейерная реализация дает выигрыш в 1.7 раза по сравнению с последовательной, что говорит о преимуществе параллельной реализации этапов конвейера при решении поставленной задачи.

В ходе выполения лабораторной работы:

- были описаны и разработаны алгоритмы этапов конвейерной обработки данных;
- были описаны и разработаны линейная и конвейерная обрабоки данных;
- был реализован каждый из описанных алгоритмов;
- по экспериментальным данным были сделаны выводы об эффективности по времени каждого из реализованных алгоритмов;
- были получены зависимости времени работы линейной и конвейерной реализаций от числа и размера заявок.

Таким образом, все поставленные задачи были выполнены, а цель достигнута.

Список литературы

[1] Manjaro — enjoy the simplicity [Электронный ресурс]. Режим доступа: https://manjaro.org/ (дата обращения: 17.10.2021).

[2] Процессор Intel® Core™ i5-8265U [Электронный ресурс]. Режим доступа: https://ark.intel.com/content/www/ru/ru/ark/products/149088/intel-core-i5-8265u-processor-6m-cache-up-to-3-90-ghz.html (дата обращения: 17.10.2021).

[3] Testing – The Go Programming Language [Электронный ресурс]. Режим доступа: https://golang.org/pkg/testing/ (дата обращения: 07.12.2021).