



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 №4 по дисциплине «Анализ Алгоритмов»

Тема Параллельные вычисления на основе нативных потоков

Студент Нисуев Н.Ф.

Группа ИУ7-52Б

Преподаватель Волкова Л. Л., Строганов Д.В.

2024 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	2
1 Входные и выходные данные	3
2 Пример работы программы	4
3 Тестирование	6
4 Исследования	7
4.1 Технические характеристики	7
4.2 Описание исследования	7
ЗАКЛЮЧЕНИЕ	9
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	10

ВВЕДЕНИЕ

Многопоточность — это способность процессора или его отдельных ядер одновременно обрабатывать несколько задач или потоков, что позволяет более эффективно использовать вычислительные ресурсы. В отличие от процессов, потоки являются более легковесными, поскольку разделяют общую память и ресурсы процесса, к которому принадлежат. Благодаря этому обмен данными между потоками осуществляется быстрее, что снижает накладные расходы и повышает производительность системы в многозадачных приложениях[1].

Цель лабораторной работы — Получить навык организации параллельных вычислений на основе нативных потоков и сравнить последовательные и параллельные вычисления с использованием нативных потоков.

Для достижения поставленной цели необходимо выполнить следующие задачи:

- описать входные и выходные данные;
- реализовать последовательную и параллельную с использованием нативных потоков загрузку *HTML*—страниц;
- выполнить сравнительный анализ алгоритмов по времени выполнения в зависимости от количества загружаемых страниц.

1 Входные и выходные данные

Входные данные:

- количество страниц для загрузки с веб-сайта `gastronom.ru`;
- режим работы — последовательный или параллельный;
- количество потоков в параллельном режиме.

Выходные данные: директория с файлами, которые содержат скачанные данные со страниц рецептов.

2 Пример работы программы

На рисунках 2.1 — 2.2 показан пример работы программы в последовательном режиме. Программа последовательно обрабатывает три страницы сайта `gastronom.ru`, сохраняя данные рецепта с каждой страницы в отдельные файлы в формате *JSON*, которые размещаются в директории *recipes_data*. Пример содержимого одного из таких *JSON*-файлов представлен на рисунке 2.5.

```
=== ПРОГРАММА ПОЛУЧАЕТ РЕЦЕПТЫ С САЙТА https://www.gastronom.ru ===  
  
Введите кол-во рецептов которое хотите считать: 3  
Введите режим считывания рецептов:  
S - Последовательный режим  
P - Параллельный режим  
: S  
  
Показать отладочную информацию? [Y/N]: Y
```

Рисунок 2.1 – Ввод данных в интерфейс приложения

```
Parsing 3 recipes  
  
Parsing ricipe[1] from URL: https://www.gastronom.ru/recipe/21234/skirli-ovsyanka-po-irlandski  
Recipe saved to recipes/skirli-ovsyanka-po-irlandski.json  
  
Parsing ricipe[2] from URL: https://www.gastronom.ru/recipe/12065/mannaya-kasha-s-chernikoj  
Recipe saved to recipes/mannaya-kasha-s-chernikoj.json  
  
Parsing ricipe[3] from URL: https://www.gastronom.ru/recipe/46595/tara-po-derbentski  
Recipe saved to recipes/tara-po-derbentski.json  
  
Parcing 3 recipes completed
```

Рисунок 2.2 – Сообщения о ходе выполнения парсинга

На рисунках 2.3 — 2.4 представлен пример работы программы в параллельном режиме. В данном случае программа создает для обработки страниц заданное количество потоков. После создания всех потоков и выдачи им задач главный поток блокируется на время парсинга всех страниц. Потоки сохраняют данные о рецепте в отдельные *JSON*-файлы, которые помещаются в директорию *recipes_data*.

```

=== ПРОГРАММА ПОЛУЧАЕТ РЕЦЕПТЫ С САЙТА https://www.gastronom.ru ===

Введите кол-во рецептов которое хотите считать: 3
Введите режим считывания рецептов:
S - Последовательный режим
P - Параллельный режим
: P
Введите кол-во потоков: 2

Показать отладочную информацию? [Y/N]: Y

```

Рисунок 2.3 – Ввод данных в интерфейс приложения

```

Parsing 3 recipes with 2 threads
Thread 1: Parsing recipe from URL: https://www.gastronom.ru/recipe/52273/olive-s-okorokom-ili-karbonadom
Thread 2: Parsing recipe from URL: https://www.gastronom.ru/recipe/57388/abrikosovoe-varene-s-kostochkami-i-limonom
Recipe saved to recipes/abrikosovoe-varene-s-kostochkami-i-limonom.json
Thread 2: Parsing recipe from URL: https://www.gastronom.ru/recipe/19928/makaronnaya-zapekanka
Recipe saved to recipes/olive-s-okorokom-ili-karbonadom.json
Thread 1 completed
Recipe saved to recipes/makaronnaya-zapekanka.json
Thread 2 completed
Parsing 3 recipes completed

```

Рисунок 2.4 – Сообщения о ходе выполнения парсинга

```

{
  "name": "Булочки из дрожжевого теста в духовке",
  "url": "https://www.gastronom.ru/recipe/33191/bulochki-iz-drozhzhevogo-testa-v-duhovke",
  "calories": "124.06 ккал/порция",
  "cook_time": "PT4H",
  "portions": "12",
  "protein": "124.06 ккал/порция",
  "fat": "6.57 г.",
  "carbohydrates": "1.01 г.",
  "ingredients": [
    "389 г хлебопекарной муки",
    "265 г воды",
    "8 г прессованных дрожжей",
    "8 г соли",
    "8 г оливкового масла",
    "1 стакан льда для создания пара"
  ]
}

```

Рисунок 2.5 – Содержимое выходного файла

3 Тестирование

В таблице 3.1 представлены функциональные тесты для разработанного программного обеспечения. Все тесты пройдены успешно.

Таблица 3.1 – Описание тестовых случаев

№	Входные данные	Ожидаемый результат	Результат теста
1	5, S	Успешная загрузка 5 страниц, сохранение в <i>recipes</i>	Пройден
2	5, P, 3	Успешная загрузка 5 страниц 2 потоками, сохранение в <i>recipes</i>	Пройден
3	a, s	Неверный ввод числа рецептов	Пройден
4	-1, s	Неверный ввод числа рецептов	Пройден
5	2, p, a	Неверный ввод числа потоков	Пройден
6	2, p, -1	Неверный ввод числа потоков	Пройден
7	Отключенное интернет-соединение	Вывод сообщений об ошибках при попытке загрузки страниц	Пройден

4 Исследования

4.1 Технические характеристики

Характеристики используемого оборудования:

- операционная система — Windows 11 Home [2]
- память — 16 Гб.
- процессор — 12th Gen Intel(R) Core(TM) i7-12700H @ 2.30 ГГц [3]

4.2 Описание исследования

Зависимости времени обработки страниц от количества страниц для последовательного и параллельных алгоритмов представлены на рисунке 4.1. Каждое значение получено путем взятия среднего из 10 измерений.

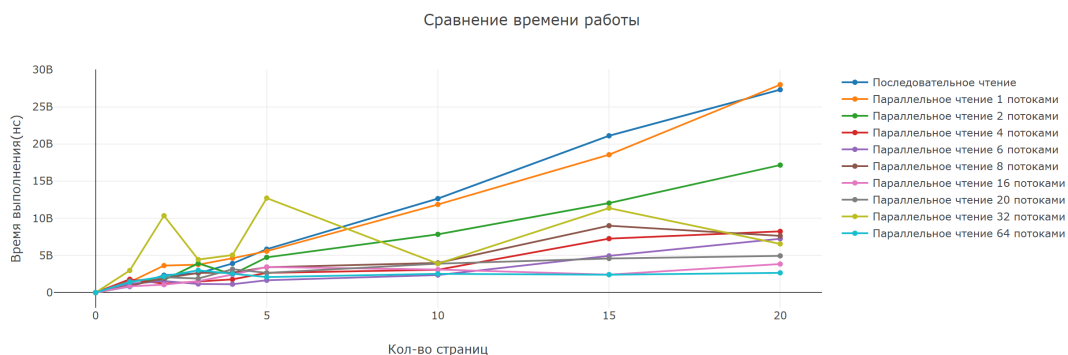


Рисунок 4.1 – График зависимости времени выполнения от количества страниц

Суммарное время загрузки всех страниц для каждого количества потоков представлено на таблице 4.1.

Таблица 4.1 – Описание тестовых случаев

Количество потоков	Суммарное время работы(нс)
0	76607837500
1	77447453100
2	51466208900
4	27502495300
6	21266895200
8	32258432800
16	18734472100
20	24755386900
32	57397960200
64	18826336700

В результате исследования было получено, что при использовании 16 потоков мы достигаем увеличение скорости в 3.2 раза относительно последовательной обработки. Также использование 16 потоков наиболее быстрое относительно других разбиений на потоки на процессоре с 12 ядрами и с 20 потоками, т.к. мы используем не все доступные ресурсы, что позволяет избежать перегрузки системы и снижает накладные расходы на управление потоками.

ЗАКЛЮЧЕНИЕ

В результате работы были реализованы алгоритмы последовательно и параллельной загрузки *HTML*—страниц, проведено сравнение времени их работы.

В ходе выполнения данной лабораторной работы были решены следующие задачи:

- описаны входные и выходные данные;
- реализованы последовательный и параллельный с использованием нативных потоков загрузку *HTML*—страниц;
- выполнен сравнительный анализ алгоритмов по времени выполнения в зависимости от количества загружаемых страниц.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Потоки и работа с ними [Электронный ресурс]. — Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/standard/threading/threads-and-threading> (дата обращения: 25.10.2024).
2. Windows technical documentation for developers and IT pros [Электронный ресурс]. — Режим доступа: <https://learn.microsoft.com/en-us/windows/> (дата обращения: 05.10.2024).
3. Intel® Core™ i7-12700H Processor [Электронный ресурс]. — Режим доступа: <https://ark.intel.com/content/www/us/en/ark/products/132228/intel-core-i7-12700h-processor-24m-cache-up-to-4-70-ghz.html> (дата обращения: 05.10.2024).