**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**Сергиенко Лев Эдуардович**

Отчет по

Лабораторная работа 6

Этапы разработки параллельных алгоритмов

|  |  | **Преподаватель** |
| --- | --- | --- |
|  | ***Кондратьева О.М.*** |
|  | | |

**2025**

# **Задание 3**

Задача – умножение матрицы на вектор.

* Разработайте схему параллельных вычислений для задачи умножения матрицы на вектор, используя методику проектирования и разработки параллельных методов.
* Опишите этапы разработки.
* Для каждого этапа ответьте на вопросы для оценки корректности выполнения этапа.

### **1. Разделение вычислений на независимые части**

**Подход:**При умножении матрицы A размера n×m на вектор x размера m каждая строка матрицы даёт один элемент результата. Таким образом, можно считать, что вычисления для каждой строки являются независимыми.

*Выполненная декомпозиция не увеличивает объем вычислений и необходимый объем памяти?*

**Ответ:** Декомпозиция по строкам матрицы не требует дополнительных вычислений – для каждой строки выполняется стандартное скалярное произведение с вектором. Дополнительная память не выделяется, за исключением хранения результатов для каждой строки.

*Возможна ли при выбранном способе декомпозиции равномерная загрузка всех имеющихся вычислительных элементов компьютерной системы?*

**Ответ:** Да. При условии равномерного распределения строк (или групп строк) между вычислительными элементами каждая задача имеет примерно одинаковую вычислительную нагрузку, что обеспечивает равномерную загрузку.

*Достаточно ли выделенных частей процесса вычислений для эффективной загрузки имеющихся вычислительных элементов (с учетом возможности увеличения их количества)?*

**Ответ:** Если число строк значительно больше числа процессоров, то каждая группа строк будет достаточно малой для балансировки нагрузки. При увеличении числа вычислительных элементов можно дополнительно делить матрицу на более мелкие подзадачи.

### **2. Выделение информационных зависимостей**

**Подход:** Каждая подзадача требует:

* Доступ к конкретной строке матрицы A (уникальные данные подзадачи).
* Доступ к вектору x (общие данные, только для чтения).
* После вычисления всех скалярных произведений необходимо собрать элементы результирующего вектора.

*Соответствует ли вычислительная сложность подзадач интенсивности их информационных взаимодействий?*

**Ответ:** Да. Каждая подзадача требует доступа к строке и общему вектору. При этом доступ к вектору осуществляется только для чтения, что соответствует низкой интенсивности информационного взаимодействия.

*Является ли одинаковой интенсивность информационных взаимодействий для разных подзадач?*

**Ответ:** Да. Все подзадачи получают одинаковый общий вектор и работают с различными строками матрицы, поэтому интенсивность обмена информацией одинакова.

*Является ли схема информационного взаимодействия локальной?*

**Ответ:** Да. Каждый вычислительный элемент работает с локальными данными и общим вектором, доступ к которому осуществляется без синхронизации, что позволяет считать схему локальной.

*Не препятствует ли выявленная информационная зависимость параллельному решению подзадач?*

**Ответ:** Нет. Информационные зависимости сведены к чтению общего вектора, что не создаёт блокировок или необходимости синхронизации между подзадачами.

### **3. Масштабирование набора подзадач**

**Подход:**Если число строк n значительно превышает число доступных процессоров p, можно объединить несколько строк в одну группу, где каждая подзадача обрабатывает n/p строк (ленточная схема). Если же строк меньше, чем вычислительных элементов, возможна дополнительная декомпозиция (например, деление строк на блоки).

*Не ухудшится ли локальность вычислений после масштабирования имеющегося набора подзадач?*

**Ответ:** Нет. При агрегации строк в группы локальность вычислений сохраняется. каждая группа обрабатывается целиком на одном процессоре.

*Имеют ли подзадачи после масштабирования одинаковую вычислительную и коммуникационную сложность?*

**Ответ:** Да. При равномерном распределении строк в группы каждая подзадача выполняет однотипные операции с минимальным обменом данными.

*Соответствует ли количество задач числу имеющихся вычислительных элементов?*

**Ответ:** Количество задач должно быть не меньше числа процессоров. При избытке задач можно применять динамическое распределение или агрегацию, чтобы каждая вычислительная единица получила хотя бы одну подзадачу.

*Зависят ли параметрически правила масштабирования от количества вычислительных элементов?*

**Ответ:** Да. Размер групп строк выбирается в зависимости от числа доступных процессоров.

### **4. Распределение подзадач между вычислительными элементами**

**Подход:**Подзадачи распределяются равномерно между доступными вычислительными элементами.

*Не приводит ли распределение нескольких задач на один процессор к росту дополнительных вычислительных затрат?*

**Ответ:** Нет. При агрегации задач распределение происходит так, что каждый процессор обрабатывает свою группу подзадач, что минимизирует накладные расходы, связанные с переключением между задачами

*Не является ли вычислительный элемент-менеджер "узким" местом при использовании схемы "менеджер-исполнитель"?*

**Ответ:** При статическом распределении задач, когда число подзадач равно числу процессоров, менеджер может отсутствовать, а распределение производится на этапе планирования. Если используется менеджер-исполнитель для динамического распределения, то это не станет узким местом.

# **Задание 4**

Задача – для файлов из заданного списка найти количество вхождений заданного слова в каждый файл.

Выполните то же, что в Задании 3.

### **1. Разделение вычислений на независимые части**

**Подход:**Основная идея - обработка каждого файла из списка является независимой. При необходимости, если файл очень большой, его можно дополнительно разбить на логические блоки (например, по строкам или фиксированным чанкам), так как подсчёт вхождений в каждом фрагменте не зависит от другого.

*Выполненная декомпозиция не увеличивает объём вычислений и необходимый объём памяти?*

**Ответ:** Обработка каждого файла или его блока проводится независимо, не добавляя избыточных вычислений или памяти, каждый файл анализируется отдельно, и данные не дублируются.

*Возможна ли при выбранном способе декомпозиции равномерная загрузка всех имеющихся вычислительных элементов компьютерной системы?*

**Ответ:** Да. При условии, что файлы (или их части) примерно одинаковы по объёму, можно распределить их между потоками. В случае неравномерности можно дополнительно разбивать большие файлы на чанки.

*Достаточно ли выделенных частей процесса вычислений для эффективной загрузки имеющихся вычислительных элементов?*

**Ответ:** Да. Если файлов много, каждое задание (файл или блок) может быть обработано отдельно. При увеличении числа вычислительных элементов можно применять динамическое распределение заданий из очереди.

### **2. Выделение информационных зависимостей**

**Подход:**Каждая подзадача получает:

* Имя файла (или блок данных) для анализа.
* Заданное слово для поиска.

Объединение результатов производится суммированием количества вхождений по каждому файлу или блоку.

*Соответствует ли вычислительная сложность подзадач интенсивности их информационных взаимодействий?*

**Ответ:** Да. Каждая подзадача проводит поиск заданного слова в своем файле / фрагменте файла - основное взаимодействие происходит на этапе сбора результатов, что соответствует относительно низкой информационной нагрузке.

*Является ли одинаковой интенсивность информационных взаимодействий для разных подзадач?*

**Ответ:** В большинстве случаев - да. Если файлы имеют схожий объём, то интенсивность взаимодействий будет одинаковой. При обработке больших и маленьких файлов возможны небольшие отличия, но их можно нивелировать на этапе агрегации.

*Является ли схема информационного взаимодействия локальной?*

**Ответ:** Да. Каждый поток работает со своим файлом (или блоком) и результат сохраняется в отдельном элементе результирующего массива, а объединение результатов производится централизованно, что не требует частых глобальных обменов.

*Не препятствует ли выявленная информационная зависимость параллельному решению подзадач?*

**Ответ:** Нет. Информационные зависимости сведены к доступу к файлам для чтения и записи результата, что не создаёт значительных препятствий для параллельного выполнения.

### **3. Масштабирование набора подзадач**

**Подход:**Если число файлов значительно больше числа вычислительных элементов, можно сгруппировать файлы в очереди заданий (или обрабатывать чанками). При наличии больших файлов- дополнительно делить файл на части, затем агрегировать результаты по файлу.

*Не ухудшится ли локальность вычислений после масштабирования имеющегося набора подзадач?*

**Ответ:** Нет. При разбиении больших файлов на блоки локальность сохраняется, так как каждая подзадача работает с конкретным участком файла.

*Имеют ли подзадачи после масштабирования одинаковую вычислительную и коммуникационную сложность?*

**Ответ:** При равномерном разбиении файлов (или при динамическом распределении блоков) подзадачи имеют схожую сложность.

*Соответствует ли количество задач числу имеющихся вычислительных элементов?*

**Ответ:** Количество задач можно регулировать: большие файлы можно разбивать на части; можно использовать пул потоков для динамического распределения, что позволит эффективно загрузить все вычислительные элементы.

*Зависят ли параметрически правила масштабирования от количества вычислительных элементов?*

**Ответ:** Размер блоков для обработки и количество одновременно выполняемых заданий подбираются с учётом числа доступных процессоров, что позволяет обеспечить оптимальное использование ресурсов.

### **4. Распределение подзадач между вычислительными элементами**

**Подход:**Задания можно распределить через пул потоков или очередь заданий, где каждый поток независимо берёт задание.

*Не приводит ли распределение нескольких задач на один процессор к росту дополнительных вычислительных затрат?*

**Ответ:** Нет. При использовании очереди заданий или пула потоков, если распределение выполнено корректно, переключение между задачами не приводит к значительным накладным расходам.

*Не является ли вычислительный элемент-менеджер "узким" местом при использовании схемы "менеджер-исполнитель"?*

**Ответ:** Если используется менеджер-исполнитель для динамического распределения заданий, то при большом количестве заданий и высокой частоте обращений следует уделить внимание оптимизации доступа к очереди, чтобы избежать узкого места.

# **Задание 5**

Задача – подсчет количества простых чисел в заданном диапазоне.

Выполните то же, что в Задании 3.

### **1. Разделение вычислений на независимые части**

**Подход:**Диапазон чисел [a,b] разбивается на несколько поддиапазонов, каждый из которых будет обрабатываться отдельно. В каждом поддиапазоне определяется, какие числа являются простыми, и производится их подсчёт.

*Выполненная декомпозиция не увеличивает объём вычислений и необходимый объём памяти?*

**Ответ:** Разбиение диапазона на поддиапазоны не добавляет лишних вычислений - проверка чисел на простоту остаётся неизменной. Дополнительная память нужна только для хранения промежуточных результатов.

*Возможна ли при выбранном способе декомпозиции равномерная загрузка всех имеющихся вычислительных элементов компьютерной системы?*

**Ответ:** Да. При условии, что диапазон делится на поддиапазоны равного фиксированного размера и каждый элемент обрабатывает равное число поддиапазонов.

*Достаточно ли выделенных частей процесса вычислений для эффективной загрузки имеющихся вычислительных элементов?*

**Ответ:** Если диапазон достаточно велик, подзадач будет много, и их можно распределить между вычислительными элементами.

### **2. Выделение информационных зависимостей**

**Подход:**Поскольку проверка чисел на простоту не требует обмена данными между подзадачами, информационные зависимости сведены к сбору результатов (суммирование количества простых чисел из каждого поддиапазона).

*Соответствует ли вычислительная сложность подзадач интенсивности их информационных взаимодействий?*

**Ответ:** Да. Каждая подзадача независимо проверяет числа в своём диапазоне, а обмен информацией происходит только на этапе суммирования результатов, что соответствует низкой интенсивности обмена.

*Является ли одинаковой интенсивность информационных взаимодействий для разных подзадач?*

**Ответ:** Да. При равномерном разбиении диапазона каждая подзадача работает независимо и обменивается только итоговыми значениями, что делает интенсивность обмена одинакова.

*Является ли схема информационного взаимодействия локальной?*

**Ответ:** Да. Каждая подзадача работает со своим поддиапазоном, а объединение результатов выполняется централизованно, что не требует частых глобальных обменов.

*Не препятствует ли выявленная информационная зависимость параллельному решению подзадач?*

**Ответ:** Нет. Слабые информационные зависимости (только сбор итогов) не создают препятствий для параллельного выполнения, так как основная нагрузка ложится на независимую проверку чисел.

### **3. Масштабирование набора подзадач**

**Подход:**Если количество поддиапазонов меньше числа вычислительных элементов, можно дополнительно декомпозировать диапазон на меньшие отрезки. Если поддиапазонов слишком много, их можно объединить для уменьшения числа задач.

*Не ухудшится ли локальность вычислений после масштабирования имеющегося набора подзадач?*

**Ответ:** Нет. Разбиение диапазона на поддиапазоны сохраняет локальность - каждая задача проверяет конкретный набор чисел без взаимодействия с другими задачами.

*Имеют ли подзадачи после масштабирования одинаковую вычислительную и коммуникационную сложность?*

**Ответ:** Да. При равномерном делении диапазона каждая подзадача содержит приблизительно равное количество чисел для проверки (хотя в зависимости от выбранного алгоритма, большие числа будут обрабатываться медленнее. Это исправляется разбиением на диапазоны фиксированного размера и динамическим распределением задач), а коммуникация ограничивается объединением результатов.

*Зависят ли параметрически правила масштабирования от количества вычислительных элементов?*

**Ответ:** Да. Размер поддиапазонов подбирается в зависимости от числа доступных процессоров, чтобы каждая вычислительная единица получила оптимальный объём работы.

### **4. Распределение подзадач между вычислительными элементами**

**Подход:**Подзадачи (поддиапазоны) распределяются равномерно между процессорами. Возможны два варианта:

* Статическое распределение, когда каждому процессору изначально назначается свой поддиапазон;
* Динамическое распределение через очередь заданий, если время обработки подзадач может значительно различаться.

*Не приводит ли распределение нескольких задач на один процессор к росту дополнительных вычислительных затрат?*

**Ответ:** Нет. Если подзадачи распределяются равномерно (например, через пул потоков или динамическую очередь), затраты на переключение минимальны, а нагрузка распределена равномерно.

*Существует ли необходимость динамической балансировки вычислений?*

**Ответ:** В случае, если некоторые подзадачи требуют большего времени, динамическое распределение поможет скорректировать нагрузку и избежать простаивания вычислительных элементов.

*Не является ли вычислительный элемент-менеджер "узким" местом при использовании схемы "менеджер-исполнитель"?*

**Ответ:** При правильной реализации менеджера данный подход не создаёт узкого места, так как обмен данными происходит лишь при выдаче заданий и сборе итоговых значений.