Практическая работа №4

Тема: Оптимизация параллельного кода на GPU с использованием различных типов памяти

Цель работы: Изучение и использование различных типов памяти CUDA (глобальная, разделяемая, локальная) для оптимизации параллельных вычислений на GPU.

Теоретическая часть

1) **Типы памяти в CUDA:**

   - **Глобальная память**: Доступна всем потокам, но медленная. Используется для хранения больших объемов данных.
   - **Разделяемая память**: Быстрая, общая для потоков одного блока. Эффективна для обмена данными между потоками внутри блока.
   - **Локальная память**: Хранится в регистре, доступна только одному потоку. Быстрая, но ограниченная по объему.

2) **Оптимизация с использованием памяти:**

   - Минимизация доступа к глобальной памяти через использование разделяемой памяти.
   - Доступ к глобальной памяти для повышения пропускной способности.
   - Использование локальной памяти для временных переменных.

3) **Пример использования:** Рассмотрим задачу параллельного нахождения суммы элементов массива. Глобальная память будет использоваться для исходного массива, а разделяемая – для временных промежуточных сумм.

Практическая часть

Задания

1) Подготовка данных:

- Реализовать программу для генерации массива случайных чисел (размер: 1,000,000 элементов).

2) Оптимизация параллельного редукционного алгоритма:

- **Реализовать редукцию суммы элементов массива с использованием:**
    a. Только глобальной памяти.
    b. Комбинации глобальной и разделяемой памяти.
- Сравнить производительность и объяснить влияние использования разделяемой памяти.

3) Оптимизация сортировки на GPU:

- Реализовать сортировку пузырьком для небольших подмассивов с использованием локальной памяти.
- Использовать глобальную память для хранения общего массива.
- Реализовать слияние отсортированных подмассивов с использованием разделяемой памяти.

4) Измерение производительности:

- Замерить время выполнения программ с использованием разных типов памяти для массивов размером 10,000, 100,000 и 1,000,000 элементов.
- Построить графики зависимости времени выполнения от размера массива.

Контрольные вопросы

1. Чем отличаются типы памяти в CUDA и в каких случаях их использовать?
2. Как использование разделяемой памяти влияет на производительность?
3. Доступ и как его обеспечить?
4. Какие сложности возникают при работе с большим объемом данных на GPU?
5. Почему важно минимизировать доступ к глобальной памяти?
6. Как использовать профилирование для анализа производительности CUDA-программ?