Курс «Архитектура вычислительных систем». СибГУТИ. 2020 г.

Лабораторная работа № 4. Оптимизация доступа к памяти.

1. На языке C/C++/C# реализовать функцию DGEMM BLAS последовательное умножение двух квадратных матриц с элементами типа double. Обеспечить возможность задавать размерности матриц в качестве аргумента командной строки при запуске программы. Инициализировать начальные значения матриц случайными числами.
2. Провести серию испытаний и построить график зависимости времени выполнения программы от объёма входных данных. Например, для квадратных матриц с числом строк/столбцов 1000, 2000, 3000, … 10000.
3. Оценить предельные размеры матриц, которые можно перемножить на вашем вычислительном устройстве.
4. Реализовать дополнительную функцию DGEMM_opt_1, в которой выполняется оптимизация доступа к памяти, за счет построчного перебора элементов обеих матриц.
5. * Реализовать дополнительную функцию DGEMM_opt_2, в которой выполняется оптимизация доступа к памяти, за счет блочного перебора элементов матриц. Обеспечить возможность задавать блока, в качестве аргумента функции.
6. ** Реализовать дополнительную функцию DGEMM_opt_3, в которой выполняется оптимизация доступа к памяти, за счет векторизации кода.
7. Оценить ускорение умножения для матриц фиксированного размера, например, 1000x1000, 2000x2000, 5000x5000, 10000x10000.
   * Для блочного умножения матриц определить размер блока, при котором достигается максимальное ускорение.
8. С помощью профилировщика для исходной программы и каждого способа оптимизации доступа к памяти оценить количество промахов при работе к КЭШ памятью (cache-misses).
9. Подготовить отчет отражающий суть, этапы и результаты проделанной работы.

Материалы для самостоятельного изучения:

Архив курса по дисциплине «Высокопроизводительные вычислительные системы» (М.Г. Курносов) URL: https://mkurnosov.net/docs/hpcs-2015-2016.tar.bz2

- Лекция 3 «Оптимизация доступа к памяти»
- Лекция 4 «Векторизация кода»