Задание:

Написать параллельную функцию, выполняющую умножение матрицы на вектор. При этом умножение должно быть реализовано следующим образом:

- для заданной матрицы A и вектора x необходимо вычислить произведение y=Ax;
- на место вектора x записать результат умножения матрицы на вектор y;

Эту последовательность действий нужно повторить 100 раз.

Для выполнения этого задания нужно скачать здесь проект MPI_Example2 и в данном проекте распараллелить функцию multvm.

Условия выполнения (инструкция располагается в архиве с проектом):

Для выполнения задания необходимо скачать проект Example2 и реализовать параллельный алгоритм вычисления произведения матрицы на вектор в рамках функции multmv (int n, double* A, double* x, double* y). В массив y(:) должен выдаваться результат выполнения операции.

К проекту прилагается dll-библиотека с процедурой Submit, в которой будет произведена корректность работы вашей функции. Для проверки задания раскомментируйте вызов функции Submit() в файле mail.cpp и запустите программу на выполнение. После завершения программы будет создан файл submit.bin, который надо загрузить на сайт Coursera для проверки задания.

В рамках проекта уже проведена инициализация MPI. Для каждого процесса определен размер подзадачи. Также в соответствии с размером подзадачи на каждом процессе выделена память под массивы соответствующей длины. В рамках процедуры параллельного умножения матрицы на вектор необходимо сначала распределить данные по процессам. Каждому процессу должна достаться часть матрицы A и целиком вектор x. Далее каждый процесс должен выполнить операцию умножения своей части матрицы A на вектор x и получить свою часть вектора y длиной n1. После этого результирующий вектор y переписывается на место вектора x. И вновь повторяется умножение матрицы на вектор.

Проект настроен на платформу **x64** и выполнение в режиме компиляции **release.**

Задание считается выполненным правильно при выполнении следующих условий:

1. При запуске программы на выполнение более чем на одном процессе результат работы пользовательской функции совпадает с результатом работы эталонной функции в dll-библиотеке.

2. Эффективность выполнения пользовательской параллельной функции более 0,6.

Замечания:

1. Эффективность - это отношение ускорения работы параллельной программы к количеству задействованных процессов. Максимальную эффективность можно получить при использовании двух или трех процессов.

2. Матрицу по процессам необходимо распределить только один раз на этапе инициализации. В то время как результирующий вектор у нужно собирать после каждой итерации (после каждого умножения матрицы на вектор).