Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Московский государственный технический университет имени Н.Э. Баумана»

(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

**Лабораторная работа №1**

**«Распараллеливание алгоритма вычисления**

**произведения двух матриц»**

**по курсу: «Разработка параллельных и распределенных программ»**

Выполнил:

Студент группы ИУ9-52Б

Виленский С. Д.

Проверил:

Царёв А. С.

**Код:**

**import numpy as np**

**import asyncio**

**import time**

**def dot\_matrices(**

**left\_matrix: np.matrix[np.int64],**

**right\_matrix : np.matrix[np.int64]**

**) -> np.matrix[np.int64]:**

**assert left\_matrix.shape == right\_matrix.shape and left\_matrix.shape[0] == left\_matrix.shape[1]**

**result\_matrix = np.matrix(np.zeros(left\_matrix.shape, dtype=np.int64))**

**for row in range(result\_matrix.shape[0]):**

**for col in range(result\_matrix.shape[1]):**

**for i in range(result\_matrix.shape[0]):**

**result\_matrix[row, col] += left\_matrix[row, i] \* right\_matrix[i, col]**

**return result\_matrix**

**async def private\_dot\_matrices\_parallel(**

**left\_matrix: np.matrix[np.int64],**

**right\_matrix : np.matrix[np.int64],**

**block\_size : np.int8,**

**block\_cords : tuple[np.int8, np.int8],**

**result\_matrix : np.matrix[np.int64]**

**) -> None:**

**for row in range(block\_cords[0], block\_cords[0] + block\_size):**

**for col in range(block\_cords[1], block\_cords[1] + block\_size):**

**for i in range(left\_matrix.shape[0]):**

**result\_matrix[row, col] += left\_matrix[row, i] \* right\_matrix[i, col]**

**async def dot\_matrices\_parallel(**

**left\_matrix: np.matrix[np.int64],**

**right\_matrix : np.matrix[np.int64],**

**split\_size : np.int64 = 1**

**) -> np.matrix[np.int64]:**

**assert left\_matrix.shape == right\_matrix.shape and left\_matrix.shape[0] == left\_matrix.shape[1]**

**assert left\_matrix.shape[0] % split\_size == 0**

**result\_matrix = np.matrix(np.zeros(left\_matrix.shape, dtype=np.int64))**

**block\_size = left\_matrix.shape[0] // split\_size**

**async with asyncio.TaskGroup() as tg:**

**for block\_row in range(0, split\_size \* block\_size, block\_size):**

**for block\_col in range(0, split\_size \* block\_size, block\_size):**

**tg.create\_task(**

**private\_dot\_matrices\_parallel(**

**left\_matrix, right\_matrix,**

**block\_size, (block\_row, block\_col),**

**result\_matrix)**

**)**

**return result\_matrix**

**def main():**

**size = 150**

**left\_matrix = np.matrix(np.random.rand(size, size) \* 10, dtype=np.int64)**

**right\_matrix = np.matrix(np.random.rand(size, size) \* 10, dtype=np.int64)**

**print("Numpy lib: ", end="")**

**start\_time = time.time()**

**result\_matrix = np.dot(left\_matrix, right\_matrix)**

**assert np.array\_equal(result\_matrix, result\_matrix)**

**print("%f\n" % (time.time() - start\_time))**

**print("Parallel hand-made func (treads: 10): ", end="")**

**start\_time = time.time()**

**assert np.array\_equal(result\_matrix, asyncio.run(dot\_matrices\_parallel(left\_matrix, right\_matrix, 10)))**

**print("%f\n" % (time.time() - start\_time))**

**print("Parallel hand-made func (treads: 5): ", end="")**

**start\_time = time.time()**

**assert np.array\_equal(result\_matrix, asyncio.run(dot\_matrices\_parallel(left\_matrix, right\_matrix, 5)))**

**print("%f\n" % (time.time() - start\_time))**

**print("Parallel hand-made func (treads: 2): ", end="")**

**start\_time = time.time()**

**assert np.array\_equal(result\_matrix, asyncio.run(dot\_matrices\_parallel(left\_matrix, right\_matrix, 2)))**

**print("%f\n" % (time.time() - start\_time))**

**print("Parallel hand-made func (treads: 1): ", end="")**

**start\_time = time.time()**

**assert np.array\_equal(result\_matrix, asyncio.run(dot\_matrices\_parallel(left\_matrix, right\_matrix, 1)))**

**print("%f\n" % (time.time() - start\_time))**

**print("Linear hand-made func: ", end="")**

**start\_time = time.time()**

**assert np.array\_equal(result\_matrix, dot\_matrices(left\_matrix, right\_matrix))**

**print("%f\n" % (time.time() - start\_time))**

**if \_\_name\_\_ == "\_\_main\_\_":**

**main()**

**Характеристики устройства:**

Процессор: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz

Оперативная память: 16,0 ГБ (доступно: 15,8 ГБ)

Тип системы: 64-разрядная операционная система, процессор x64

**Вывод программы:**

Numpy lib: 0.003608

Parallel hand-made func (treads: 10): 5.315130

Parallel hand-made func (treads: 5): 5.140474

Parallel hand-made func (treads: 2): 4.434812

Parallel hand-made func (treads: 1): 5.426144

Linear hand-made func: 5.316972

**Вывод:**

В ходе выполнения лабораторной работы была использована библиотека asyncio языка Python. Как мы можем заметить при разных количествах потоков результаты затраченного времени при распараллеливании и без него изменяются в пределах погрешности, из чего можно сделать вывод, что данная библиотека не подходит для разработки многопоточных программ.