Задание 3
Подсчёт количества промахов в кэш для операции матричного умножения в зависимости от порядка итерирования.
Отчёт

Кучеров В.Д.

2022

1. Постановка задачи

При помощи PAPI снять значения аппаратных счетчиков промахов L1/L2 кэшей при выполнении оперции умножения квадратных матриц. Сравнить полученные значения с теоретическими для каждого порядка итерирования.

2. Реализация

В ходе работы подготовлено две программы: **square_matrix_generator** и **papi_matrix_mul** для настраивомой случайной генерации квадратных матриц и перемножения матриц соответственно. Проверка корректности перемножения матриц производилась вручную с использованием стороннего ресурса - `https://www.wolframalpha.com/` .Формула для умножения матриц:

$$c_{ij} = \Sigma_{k=1}^{n} a_{ik} \cdot b_{kj}, \quad i, j = 1, 2, ..., n.$$

Для подсчета количества промахов кэша использовалась библиотека *PAPI*, а именно события **PAPI_L1_DCM** (Level 1 data cache misses) и **PAPI_L2_DCM** (Level 2 data cache misses)

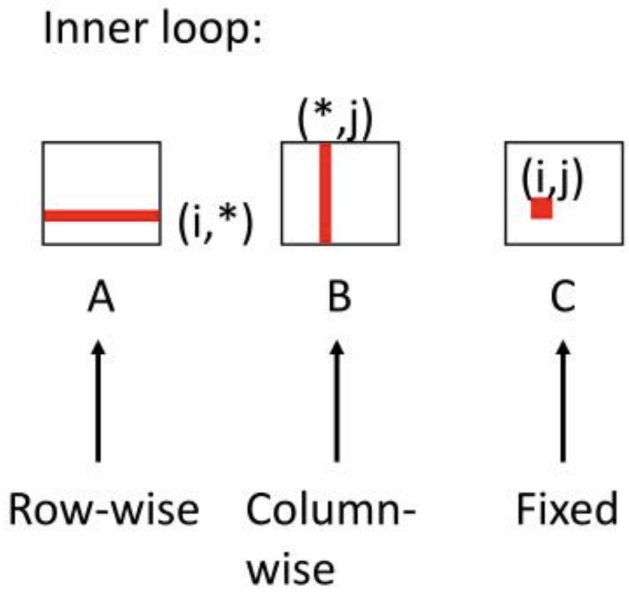
3. Теоретический подсчет промахов кэша L1

1. **ijk (jik)**

   Доступ к элементам матрицы A происходит построчно, в кэш записывается $\frac{\text{cache\_line\_size}}{\text{element\_size}}$ элементов, в проведенном исследовании $cache\_line\_size = 128$ байт, $element\_size = 4$ байт, соответственно в одну линию кэша помещается $\frac{128}{4} = 32$ элемента. Значит промах кэша будет происходить на каждый 32-й доступ к элементу матрицы A. Доступ к элементам матрицы B происходит с шагом N - длина строки матрицы, соответственно при больших N промах кэша будет происходить при каждом чтении элемента B. Для фиксированного элемента матрицы C заведена временная переменная, поэтому промахов кэша не будет. Случай **jik** аналогичен, только A и B меняются местами, но суммарное число промахов не изменится.

	A	B	C	Всего
Количество промахов на одну итерацию (ijk)	0,03125	1,00	0,00	1,03125
Количество промахов на одну итерацию (jik)	1,00	0,03125	0,00	1,03125

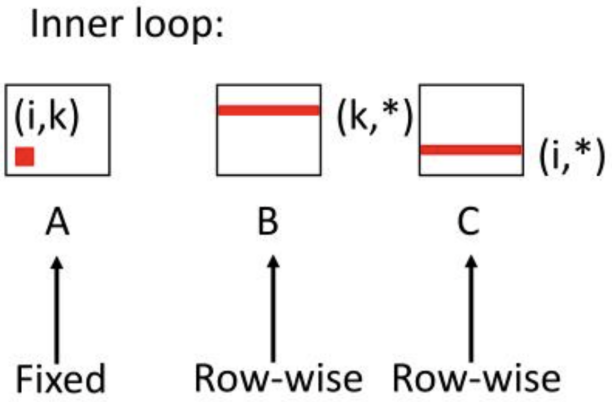
   В случае матрицы размера 1000 общее число промахов будет $1,03125 * N^3 = 1,03125 * 1000^3 = 1,03125e9$



2. **ikj (kij)** Фиксированный элемент матрицы A, и построчный доступ к элементам матриц B и C. Соответственно, обращений к элементам матрицы A не будет во внутреннем цикле, а промахи кэша для B и C считаются так же как и в первом случае: $\frac{\text{cache\_line\_size}}{\text{element\_size}} = \frac{128}{4} = 32$, а значит промахи будут при каждом 32-ом обращении к массивам B и C.

	A	B	C	Всего
Количество промахов на одну итерацию (ikj)	0	0,03125	0,03125	0,0625
Количество промахов на одну итерацию (kij)	0	0,03125	0,03125	0,0625

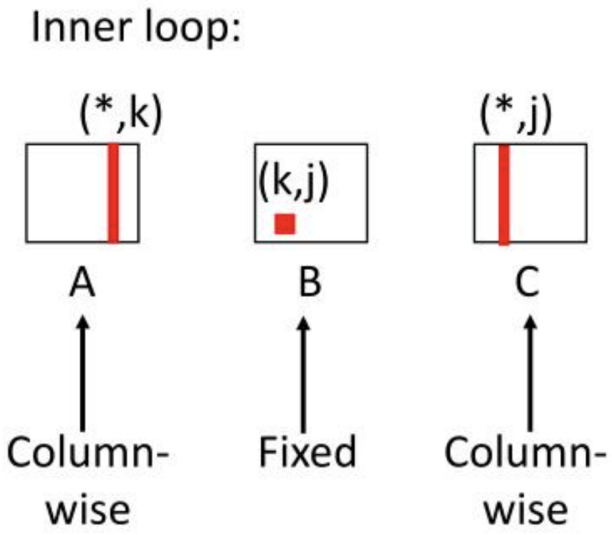
В случае матрицы размера 1000 общее число промахов будет $0,0625 * N^3 = 0,0625 * 1000^3 = 6,25e7$



3. **jki (kji)** Фиксированный элемент матрицы B и обращение по столбцам к матрицам A и C. Для матрицы B промахов не будет, а для матриц A и C при больших N промах кэша будет при кажом обращении к элементам этих матриц.

	A	B	C	Всего
Количество промахов на одну итерацию (jki)	1	0	1	2
Количество промахов на одну итерацию (kji)	1	0	1	2

В случае матрицы размера 1000 общее число промахов будет $2*N^3 = 2*1000^3 = 2e9$



4. Формат командной строки

```
./papi_matrix_mul <файл матрицы a> <файл матрицы b> <файл матрицы c> <режим>
```

режим: выбор порядка итерирования:

0. ijk
1. ikj
2. kij
3. jik
4. jki
5. kji

```
./square_matrix_generator  <количество строк/столбцов> <имя выходного файла>
                           <опц. зерно генерации> <опц. максимальное значение>
                           <опц. генерировать отрицательные числа>
```

1. **зерно генерации**: позволяет генерировать одинаковые массивы

2. **максимальное значение**: максимальное число, которое может быть сгенерировано (по умолчанию **RAND_MAX**)

3. **генерировать отрицательные числа**: 1 (значение по умолчанию) - генерировать отрицательные числа (максимальное значние будет при этом уменьшено вдвое), 0 - запретить генерацию отрицательных чисел

5. Спецификация системы

POLUS
Модель процессора: IBM POWER8
Размер линии кэша L1: 128 bytes
Размер кэша L1: 64K
Размер кэша L2: 512K

6. Результаты выполнения

Режим	L1 Cache misses	L2 Cache misses	L1 theor	L1 theor / L1 Cache misses
0 (ijk)	1,66E+09	3,53E+07	1,03E+09	0,62
1 (ikj)	2,33E+07	6,59E+05	6,25E+07	2,68
2 (kij)	2,76E+07	1,72E+06	6,25E+07	2,26
3 (jik)	1,66E+09	3,30E+07	1,03E+09	0,62
4 (jki)	3,19E+09	1,26E+08	2,00E+09	0,63
5 (kji)	3,19E+09	1,26E+08	2,00E+09	0,63