Московский Государственный Университет им. М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Суперкомпьютеров и Квантовой Информатики

Спецкурс: системы и средства параллельного программирования

Отчёт № 2

Анализ влияния кэша на операцию блочного матричного умножения

Работу выполнила
Домрачева Д. А.

Москва 2018

Постановка задачи и формат данных

Задача:

Реализовать последовательный алгоритм матричного умножения и оценить влияние кэша на время выполнения программы.

Формат командной строки:

<имя файла матрицы A> <имя файла матрицы B> <имя файла матрицы C> <размер блока> <порядок индексов> <счетчики>.

Параметры командной строки, определяющие режим выполнения:

- $0 - ijk$, $1 - ikj$;
- 32 – блок размера 32×32, или 0 – блок оптимального размера, вычисляемый по формуле: $3 \times b^2 = m \times L$, где b – размер блока в элементах, а $m \times L$ – размер кэша;
  Примечание: на устройстве, где осуществлялся запуск программы, размер блока равняется 52.
- 1, 2, 3, 4, 5, 6, 7 – режимы для вычисления определенных параметров системы, могут быть заданы в произвольном порядке, где, соответственно:
  - 1 – PAPI_L1_TCM – промахи кэша L1;
  - 2 – PAPI_L2_TCM – промахи кэша L2;
  - 3 – PAPI_L3_TCM – промахи кэша L3;
  - 4 – PAPI_TOT_INS – общее число выполненных инструкций;
  - 5 – PAPI_TLB_IM – промахи инструкций TLB;
  - 6 – PAPI_TOT_CYC – общее число циклов;
  - 7 – PAPI_TLB_DM – промахи данных TLB.

Примечание: среди доступных счетчиков на устройстве, где выполнялись запуски программы, нет счетчиков для определения числа FLOP.

Формат файла с матрицей:

Матрица представляются в виде бинарного файла следующего формата:

Тип	Значение	Описание
Число типа *size_t*	N – натуральное число	Число строк матрицы
Число типа *size_t*	M – натуральное число	Число столбцов матрицы
Массив чисел типа T	$N \times M$ элементов	Массив элементов матрицы

Элементы матрицы хранятся построчно.

Описание алгоритма

Математическая постановка:

Алгоритм матричного умножения $(A \times B = C)$ можно представить в следующем виде:

$$c_{ij} = \sum_{\substack{k=1 \\ k+=s}}^{n} \sum_{k_1=k}^{k+b} \left(a_{ik_1} \cdot b_{k_1 j} \right)$$ для каждого элемента матрицы C.

Оценка влияния кэша на время выполнения программы осуществляется за счёт перестановки индексов суммирования и за счет изменения размеров блока.

Анализ выполнения:

Для оценки времени выполнения программы использовалась функция *clock()*. Для оценки таких параметров, как промахи кэша, промахи TLB и количество процессорных тактов, использовались счетчики библиотеки papi.h .

Верификация:

Для проверки корректности работы программы использовались тестовые данные. Для нахождения среднего времени выполнения умножения матриц программа запускалась по 5 раз.

Основные функции:

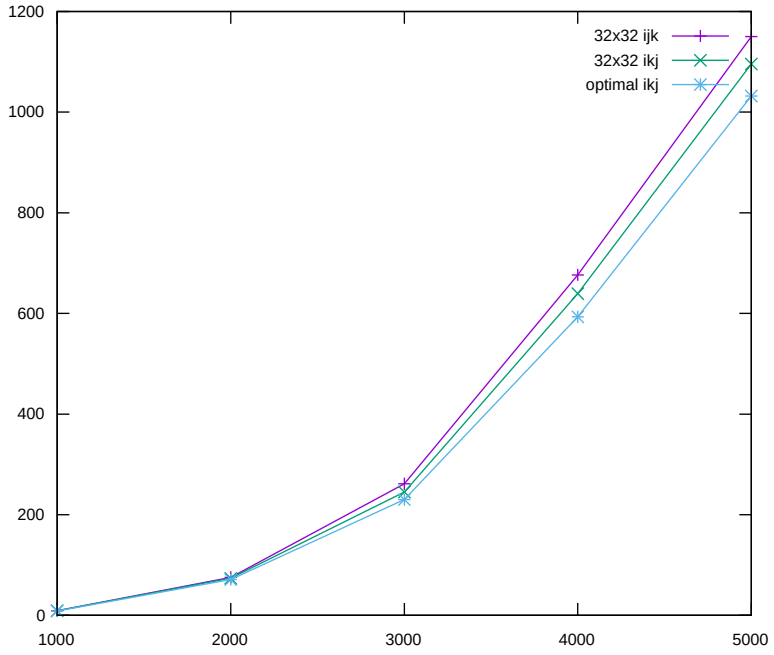
- Выделение памяти для матриц – *init_matrix*. Чтение матриц из бинарных файлов – *read_marix*. Функция для вывода матриц в файл – *write_matrix*.
- Основная функция, определяющая время выполнения умножения – *get_time*. Вызывает дополнительную функцию, реализующую умножение – *multiply_ijk* или *multiply_ikj*, в зависимости от заданного режима перемножения.
- Функции, реализовывающие умножение матриц в зависимости от порядка индексов и размера блоков – *multiply_ijk* и *multiply_ikj*. Возвращают количество тактов, за время которых выполнялись операции.

Результаты выполнения

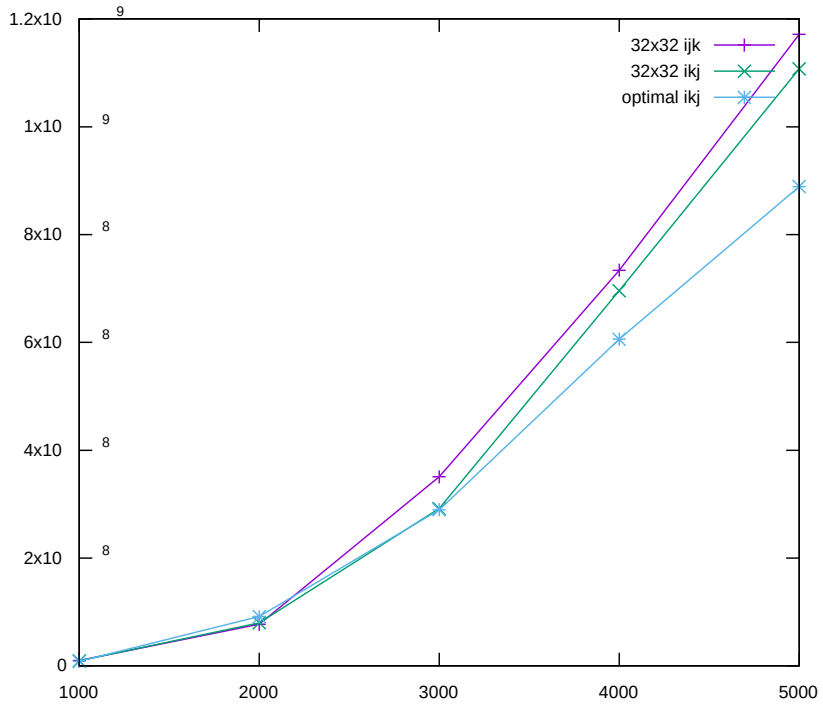
Результаты:

Проводилось перемножение матриц размерами 1000×1000 и 1000×1000, 2000×2000 и 2000×2000, 3000×3000 и 3000×3000, 4000×4000 и 4000×4000, 5000×5000 и 5000×5000. Графики показывают средние значения для 5 запусков программы.

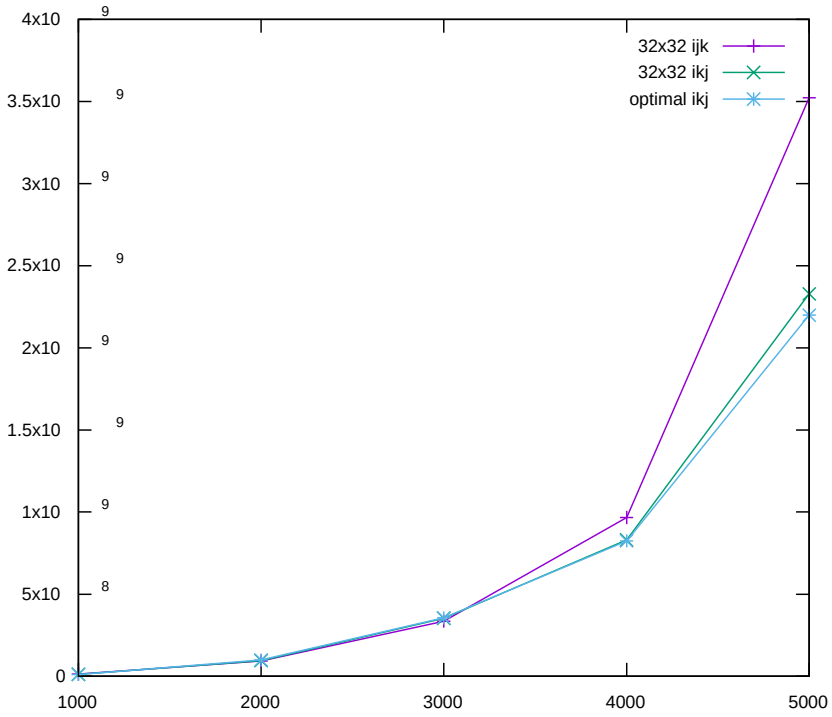
Зависимость среднего времени выполнения от порядка индексов суммирования и размера блоков представлена на графике (время в секундах):



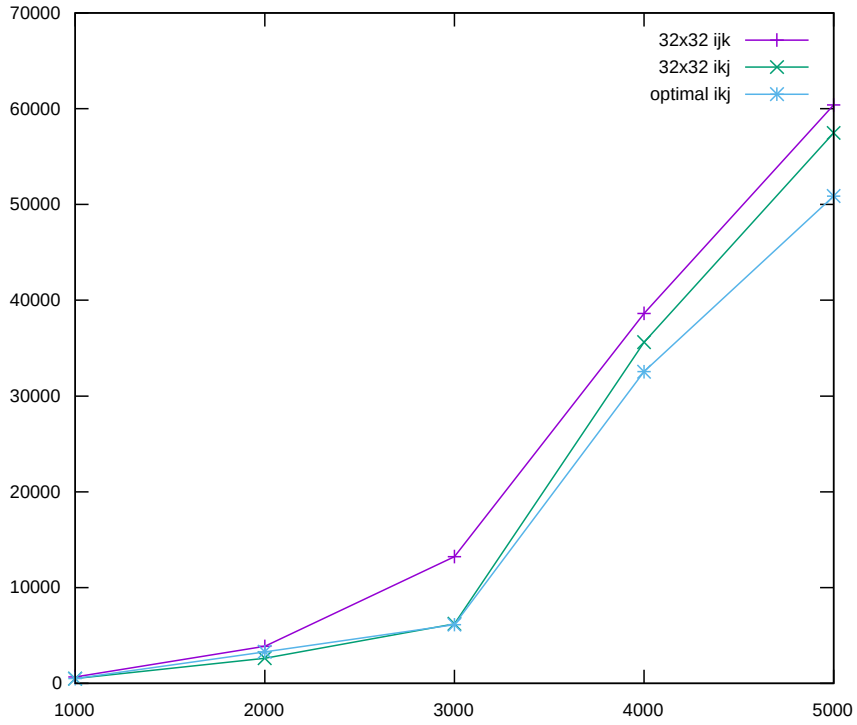
Зависимость промахов кэша L1:



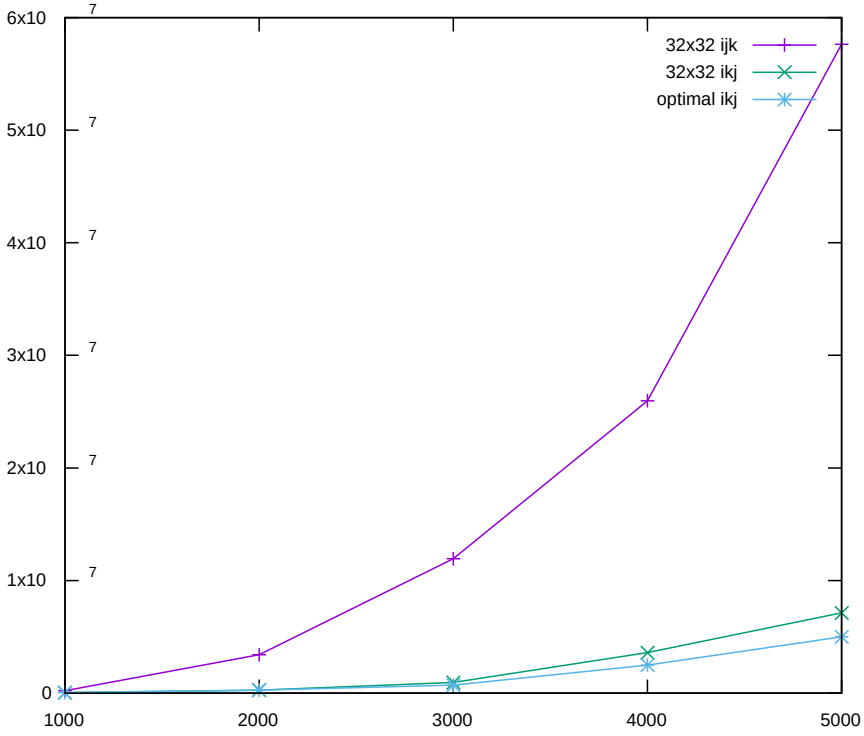
Зависимость промахов кэша L2:



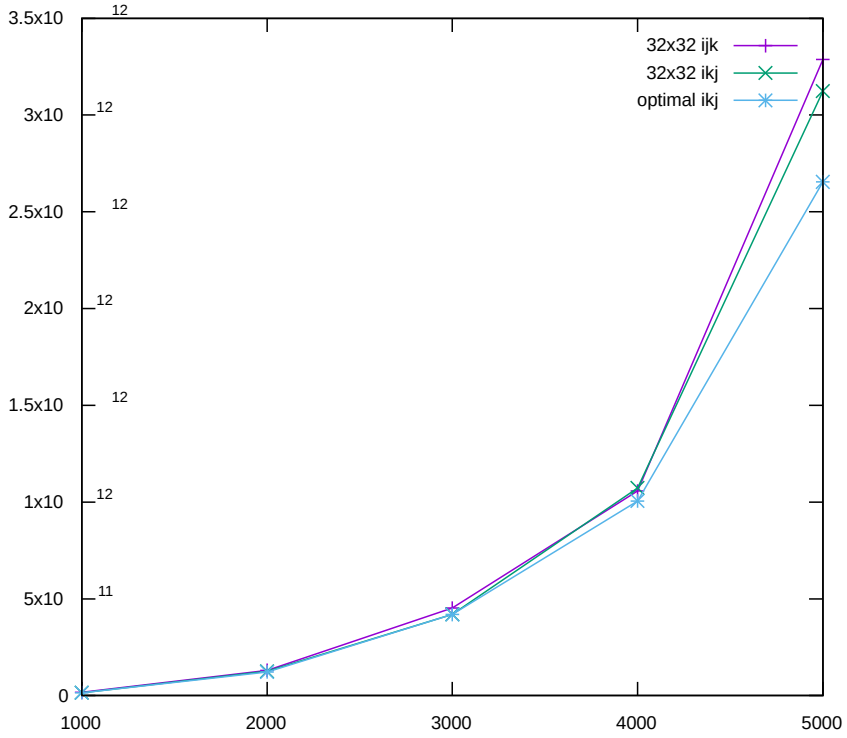
Зависимость промахов инструкций TLB:



Зависимость промахов данных TLB:



Зависимость количества тактов процессора:



Основные выводы

Основываясь на представленных графиках зависимостей промахов кэша L1 и L2, TLB, времени выполнения и числа тактов процессора, можно сделать вывод, что с ростом размеров матрицы показатели счетчиков для программы с оптимальным размером блока и порядком индексов ikj растут медленнее, чем для программ с размером блока 32 и порядками индексов ijk и ikj.