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
Оценка влияния кэша на время выполнения последовательного алгоритма блочного матричного умножения

Работу выполнил
Чепурнов А. В.

Москва 2018

Постановка задачи и формат данных

Задача: Реализовать последовательный алгоритм блочного матричного умножения и оценить влияние кэша на время выполнения программы. Воспользоваться системой PAPI для сбора информации с аппаратных счётчиков.
Формат командной строки: <имя файла матрицы A > <имя файла матрицы B > <имя файла матрицы C > <режим, порядок индексов> <размер блока, натуральное число>
Режимы: 0 – ijk, 1 – ikj
Размер блока по умолчанию: 32×32
Формат файла-матрицы: Матрица представляются в виде бинарного файла следующего формата:

Тип	Значение	Описание
Число типа char	'f' – тип float	Тип элементов матрицы
Число типа int	N – натуральное число	Число строк матрицы
Число типа int	M – натуральное число	Число столбцов матрицы
Массив чисел типа float	$N \times M$ элементов	Массив элементов матрицы

Описание алгоритма

Математическая постановка: Алгоритм блочного матричного умножения заключается в блочном представлении матриц $A\ B\ C$. Все блоки имеют фиксированный размер $S \times S$. Если $N\%S \neq 0$, то блоки последней строки имеют размер $N\%S \times S$. Если $M\%S \neq 0$, то блоки последнего столбца имеют размер $S \times M\%S$. Если выполнены оба условия, то нижний правый блок имеет размер $N\%S \times M\%S$. Умножение $A \times B$ и перемножение отдельных блоков осуществляется классическим последовательным алгоритмом матричного умножения. Оценка влияния кэша на время выполнения программы осуществляется за счёт изменения размера блоков и перестановки индексов суммирования. Формула определения оптимального блока для кэша:

$$S = \sqrt{\frac{size\_of\_cache}{3 * size\_of\_float}}$$

Аппаратное обеспечение: Вычисления проводились на отдельном ядре вычислительного комплекса IBM Polus.
L1d cache = 64K (по формуле оптимальный размер блока равен 73)
L2 cache = 512K (по формуле оптимальный размер блока равен 209)
Анализ времени выполнения: Для оценки времени выполнения программы использовалась функция: clock().
Анализ информации с аппаратных счётчиков: Для оценки влияния кэша использовались следующие средства системы PAPI:

- PAPI_L1_DCM (промахи кэша 1 уровня)
- PAPI_L2_DCM (промахи кэша 2 уровня)
- PAPI_TOT_CYC (число процессорных тактов)
- PAPI_FP_OPS (число операций с плавающей точкой)

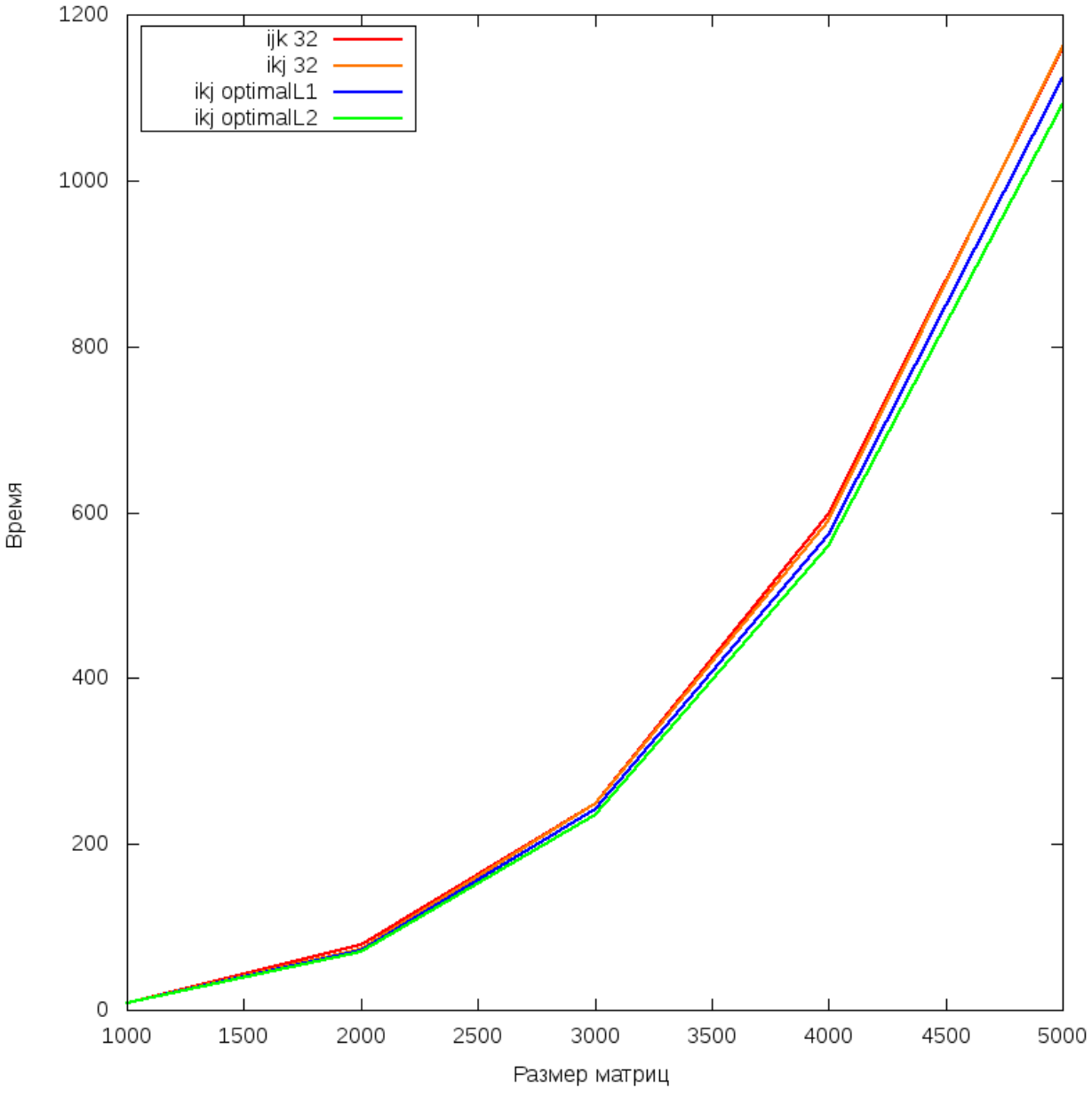
Возможности считывать информацию о буфере ассоциативной трансляции (TLB) на данном аппаратном обеспечении не было.
Верификация: Для проверки корректности работы программы использовались тестовые данные.

Основные функции:

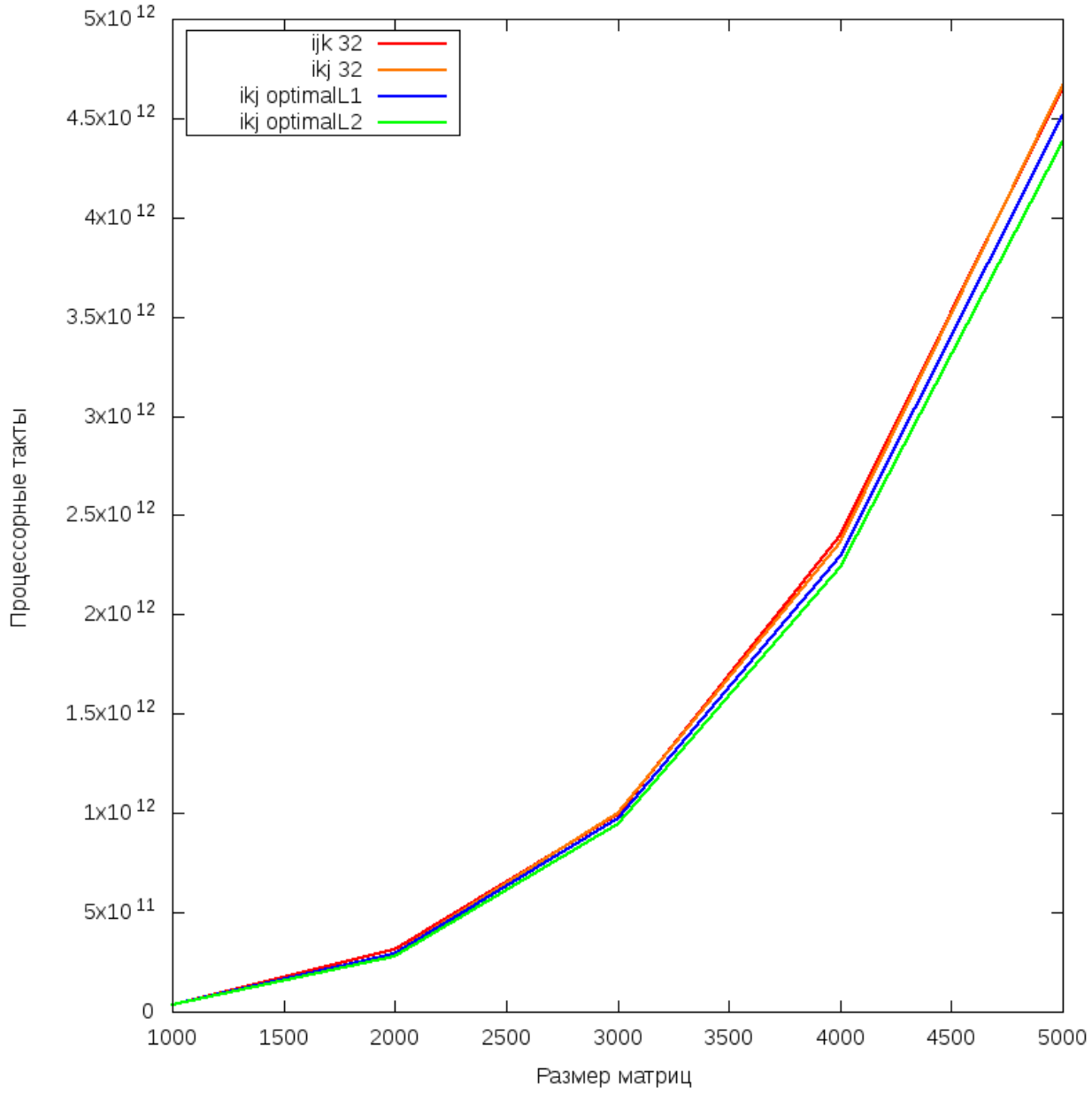
- **Разбор командной строки.** В рамках функции осуществляется анализ и разбор командной строки.
- **Чтение файлов матриц.** В рамках функции осуществляется анализ совместимости входных матриц и их чтение.
- **Перемножение матриц.** В рамках функции осуществляется перемножение матриц в соответствие с размерами блоков и порядком индексов суммирования.
- **Сбор информации о работе.** В рамках функции осуществляется подсчёт времени работы и считывание информации с аппаратных счётчиков с использованием системы PAPI.

Результаты выполнения

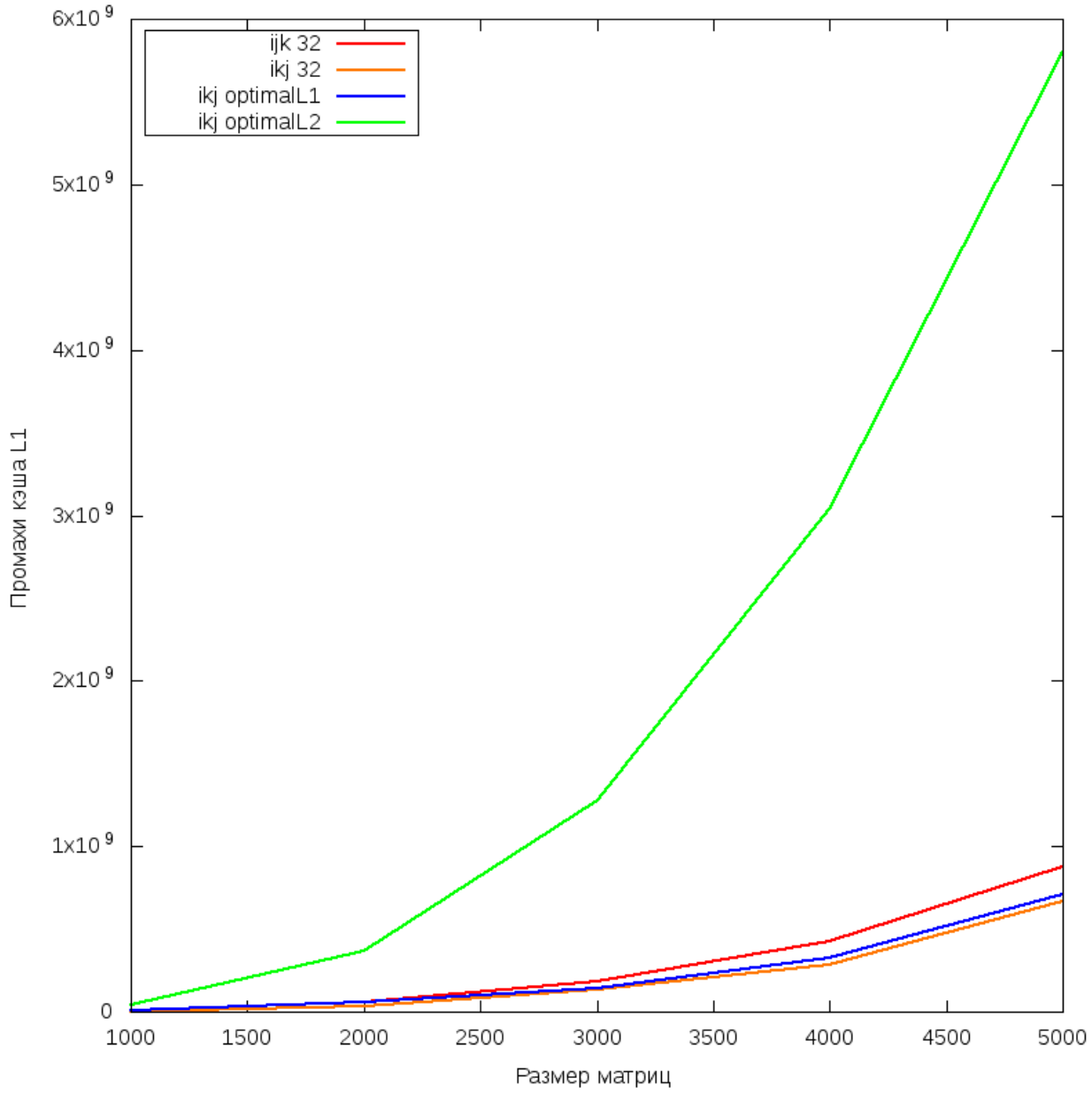
Проводилось перемножение квадратных матриц размерами 1000×1000, 2000×2000, 3000×3000, 4000×4000 и 5000×5000. Зависимость времени выполнения и числа процессорных тактов от размеров матриц представлена на графиках (время в секундах).



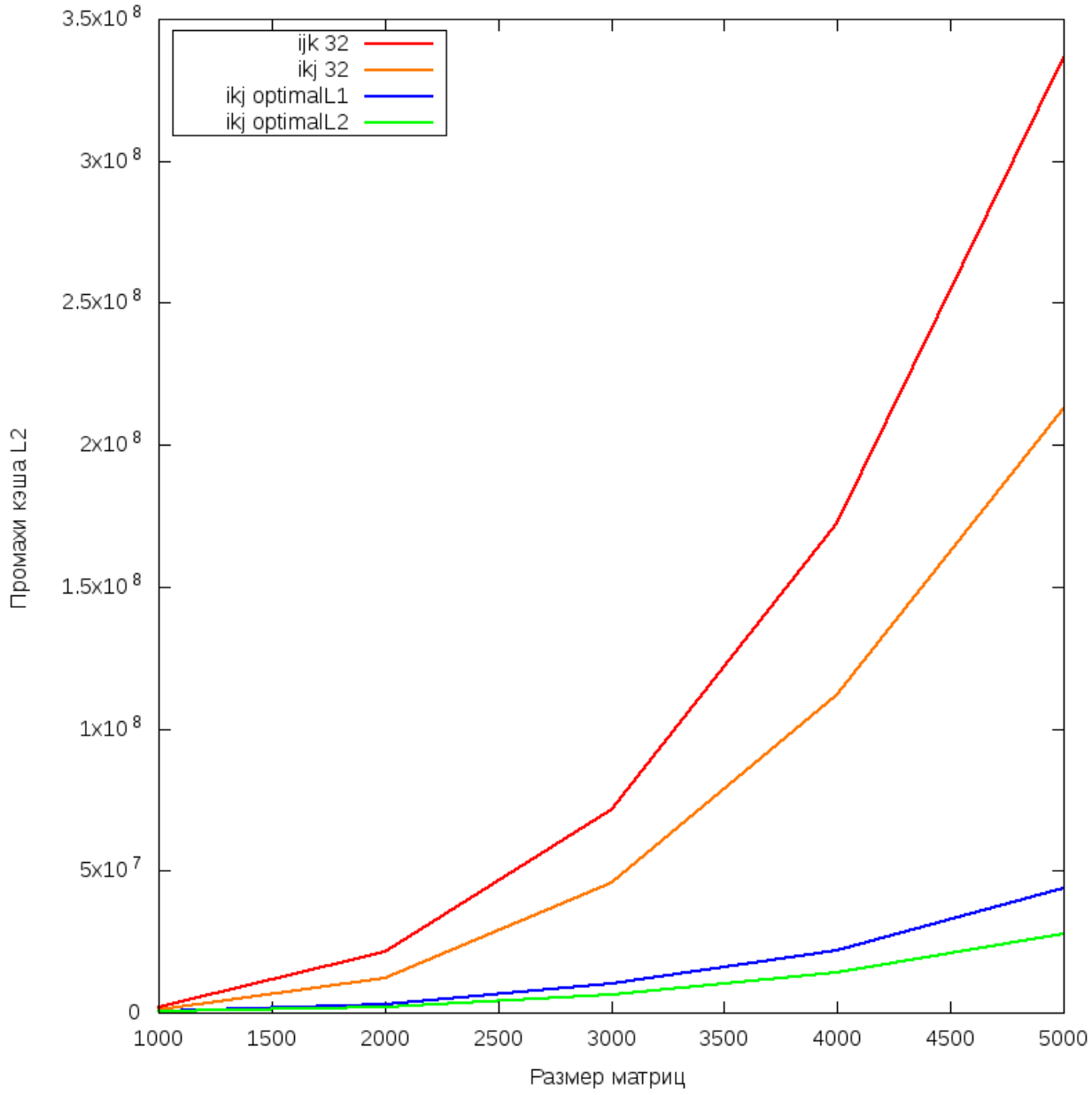
Зависимость числа процессорных тактов от размеров матриц.



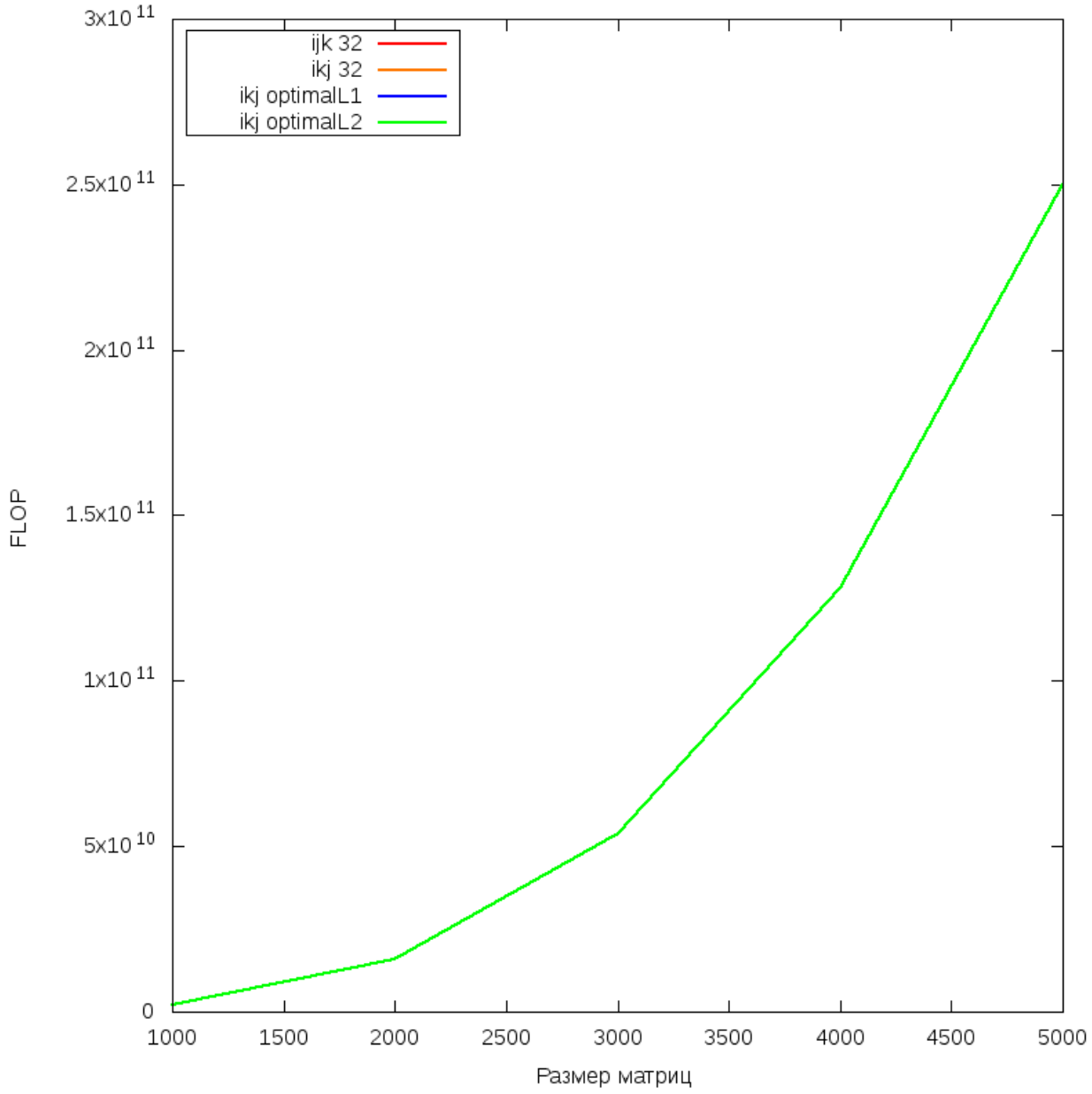
Зависимость количества промахов кэша 1 уровня от размеров матриц.



Зависимость количества промахов кэша 2 уровня от размеров матриц.



Зависимость числа FLOP от размеров матриц.



Основные выводы

Исследования показывают, что наименьшее время выполнения при оптимальном размере блока для кэша 2-го уровня (209) и порядке индексов ikj. При таком порядке весь блок умещается в кэше 2-го уровня и число его промахов снижается. Хотя число промахов кэша 1-го уровня при таких параметрах существенно больше, чем при других, общее время работы алгоритма заметно снижается (примерно на 1 минуту для матриц 5000×5000).

Следующий по скорости результат при оптимальном размере блока для кэша 1-го уровня (73) и порядке индексов ikj.

Наихудшее время при размере блока 32×32. Причем для порядка ikj все параметры лучше, чем для ijk, так как при таком порядке доступ к элементам обеих входных матриц осуществляется последовательно.

Число FLOP для каждого размера матриц при всех запусках алгоритма примерно одинаково.