###### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

###### Факультет информационных технологий

**Кафедра параллельных вычислений**

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

«ИЗМЕРЕНИЕ СТЕПЕНИ АССОЦИАТИВНОСТИ КЭШ-ПАМЯТИ»

студента 2 курса, группы 23201

Смирнова Гордея Андреевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Матвеев Алексей Сергеевич

Новосибирск 2024

1. **Цель**

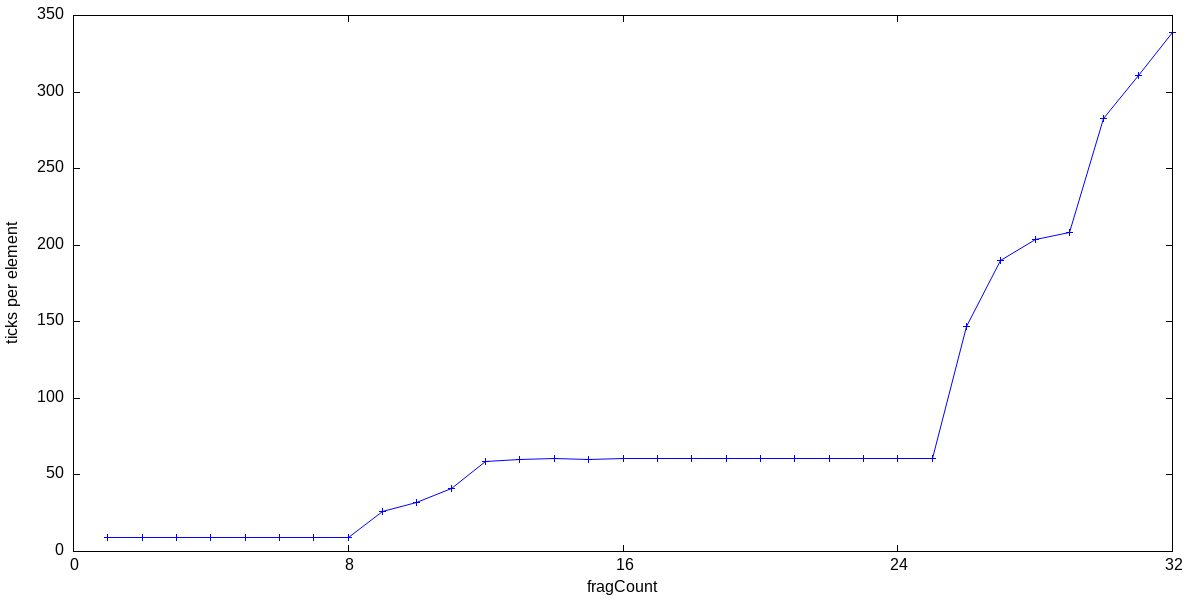
Экспериментально определить степени ассоциативности кэш-памяти разных уровней.

1. **Задание**

На основе обхода циклического массива, что вызывает кэш-буксование, сделать среднюю оценку количества тактов для обращения к одному элементу массива. На основе полученных данных выяснить степени ассоциативности кэш-памяти процессора.

1. **График роста тактов обращения**

В лабораторной работе создавался массив с фрагментами со смещением в 16МБ. Следующий график показывает зависимость времени обращения к одному элементу в тактах от количества фрагментов в массиве:



1. **Листинг программы**

#include <cstdio>

#include <climits>

#include <cmath>

#include <vector>

#include <x86intrin.h>

#define CACHE\_SIZE (32 \* 1024)

#define ITERATIONS 64

typedef unsigned long long ull;

int\* initArray(int fragCount, int offset, int arrSize) {

const auto array = new int[offset \* fragCount];

for (int i = 0; i < arrSize / fragCount; ++i) {

for (int j = 0; j < fragCount - 1; ++j) {

array[i + j \* offset] = i + (j + 1) \* offset;

}

array[i + (fragCount - 1) \* offset] = (i + 1) % (arrSize / fragCount);

}

return array;

}

ull traverseArray(const int \*array, int fragCount) {

ull minTicks = ULLONG\_MAX;

for (size\_t j = 0; j < ITERATIONS; ++j) {

const ull start = \_\_rdtsc();

for (volatile size\_t k = 0, i = 0; i < fragCount; ++i) {

k = array[k];

}

const ull end = \_\_rdtsc();

minTicks = std::min(minTicks, end - start);

}

return minTicks / fragCount;

}

void computeTicks(int fragCount, int offset, int arrSize) {

const int\* array = initArray(fragCount, offset, arrSize);

const ull ticks = traverseArray(array, arrSize);

delete[] array;

printf("%d %llu\n", fragCount, ticks);

}

int main() {

constexpr int offset = 16 \* 1024 \* 1024 / sizeof(int);

constexpr int arrSize = CACHE\_SIZE / sizeof(int);

FILE \*outputFile = freopen("output.txt", "w", stdout);

for (int fragCount = 1; fragCount <= 32; ++fragCount) {

computeTicks(fragCount, offset, arrSize);

}

fclose(outputFile);

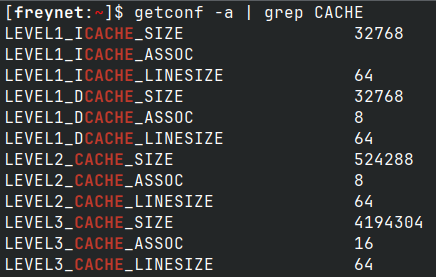
return 0;

}

1. **Оценка ассоциативности кэша**

На графике заметно два скачка. Первый скачок происходит после 8 фрагментов, что говорит об ассоциативности 8 кэша L1. Далее на 16 фрагментах скачка не происходит, поскольку на этом моменте заполняется 8-ассоциативный кэш L2, при этом кэш L3 (ввиду особенностей архитектуры Zen2) является victim-кэшем для L2 (суть в том, что при исключении элемента из L2 он не удаляется из кэша полностью, а сразу же записывается в L3). Степень ассоциативности кэша L3 – 16, однако в данной архитектуре этот кэш делится на несколько CCX, поэтому в случае процессора Ryzen 5 PRO 4650U имеем два CCX и поэтому в пределах одного блока ядер будет также ассоциативность 8. Поэтому после 24 фрагментов наблюдается заметное ухудшение производительности.

Информация об ассоциативности, полученная из конфигураций:



1. **Заключение**

В ходе данной работы было изучено строение множественно-ассоциативного кэша и эксперементально получены степени ассоциативности кэш памяти.