САНКТ – ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Архитектура ЭВМ

Отчет

По домашней работе № 3

«Кэш – память»

Выполнил: Зайнидинов Мирзофирдавс Шавкатович

Студент группы M313D

Санкт – Петербург

2020

Цель работы: закрепление материала по теме «кэш – память» путем решения задач по данной теме.

Условие задачи

Вариант № 4. Имеем следующий фрагмент кода:

```
struct element
{
double x, y, ax, ay, vx, vy, a, b;
};
void f (element arr [], int n, double asqr)
{
   for (int i = 0; i < n; ++i)
   {
      arr[i].x += arr[i].vx * asqr + 0.5 * arr[i].ax * asqr * asqr;
   }
   for (int i = 0; i < n; ++i)
   {
      arr[i].y += arr[i].vy * asqr + 0.5 * arr[i].ay * asqr * asqr;
   }
}
```

Функция f выполняется для массива из 1000 элементов в системе с кэшем данных L1 размером 32 КБ и 4 – way ассоциативностью. Размер блока составляет 64 байта. Предположим, что массив arr выровнен по адресу, кратному к 64.

Необходимо определить количество кэш – промахов и процент промахов (число промахов к общему числу обращений).

В ответе нужно представить два числа: количество кэш – промахов и % промахов.

Практическая часть

Дан кэш данных L1 размером 32 КБ и ассоциативностью 4 – way. Если размер блока 64 байта, то сначала подсчитаем количество таких блоков. Количество блоков ровняется: размер кэша деленного на размер одного блока.

32 КБ = 32 * (2 ^ 10) байт, тогда количество равен (32 * (2 ^ 10)) / 64 = 512.

Таким образом в L1 есть 128 групп по 4 строки длинной 64 байта.

Тип наших переменных double, и он занимает место 8 байт и 8 байт отводится для него в нашей строке, тогда один наш блок может вместить ровно один элемент нашего массива.

Рассмотрим первый цикл нашей программы. В начале наша кэш – память пуста. Процессор делает обращение 3 для чтения, 1 для записи. Из обращений которая делает процессор, первый будет промахом, после промаха, данные копируется в одну из кэш – линий. Итак, после каждой итерации у нас будет 1 промах из 4. В итоге после 1000 итераций происходит 4000 обращений, из которых 1000 будет промахом.

Теперь рассмотрим второй цикл нашей программы. Можно сказать, что соотношения промахов к обращениям будет как в первом цикле, однако из – за оставшихся данных, которые были записаны из первого цикла можно смело утверждать, что некоторые данные уже в кэше и при обращении они не дадут промаха.

На первой итерации, когда i станет 511 в кэш – памяти уже будет хранится элементы [0, 511] и когда будет обращение к 512, L1 будет переполнена, и не сможет хранить в себе следующие данные, после чего алгоритм вытеснения LRU сбросит начальные данные и на нее месте будет записывать новые. После первого цикла в нашей кэш памяти будут данные [488, 999].

В следующем цикле после переполнения, алгоритм будет сбрасывать данные, оставшейся элементы всегда будут сбрасываться и обновляться новыми, и в последующей итерации, в кэш – памяти не будет хранится такой элемент, который был в прошлой итерации. Соотношения промахов и обращений второго цикла будет эквивалентно первому. И в втором цикле после 4000 обращений к данным, будет 1000 промахов.

В сумме, после 8000 обращений будет 2000 промахов, которые будут равны 25%

Ответ: 2000 промахов, 25%