**CS61C Discussion 9**

**N-Way Set Associative**

Here’s some practice involving a 2-way set associative cache, similar to last discussion’s problem, except we have an 8-bit address space, classify each of the following accesses as a cache hit (H), cache miss (M), or cache miss with replacement (R). For any misses, list out which type of miss it is.

What is the T:I:O for this cache? T: 4  I: 1  O: 3

<table>
<thead>
<tr>
<th>CPU Cache</th>
<th>Index Number</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7 6 5 4 3 2 1 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. 0b0000 0100, M - Compulsory
2. 0b0000 0101, H
3. 0b0110 1000, M - Compulsory
4. 0b1100 1000, M - Compulsory
5. 0b0110 1000, H
6. 0b1101 1101, R - Compulsory
7. 0b0100 0101, M - Compulsory
8. 0b0000 0100, H
9. 0b1100 1000, R - Capacity

What is the hit rate of our 9 accesses? 3/9 or 1/3

**Analyzing C Code**

```c
#define NUM_INTS 8192
int A[NUM_INTS]; // A lives at 0x10000
int i, total = 0;
for (i = 0; i < NUM_INTS; i += 128) {A[i] = i;} // Line 1
for (i = 0; i < NUM_INTS; i += 128) {total += A[i];} // Line 2
```

Let’s say you have a byte-addressed computer with a total memory of 1 MiB. It features a 16 KiB CPU cache with 1 KiB blocks.

1. How many bits make up a memory address on this computer? **20**
2. What is the T:I:O breakdown? Tag bits: **6**  Index bits: **4**  Offset bits: **10**
3. Calculate the cache hit rate for the line marked Line 1: **50%**
   The integer accesses are 4 * 128 = 512 bytes apart, which means there are 2 accesses per block. The first accesses in each block is a compulsory cache miss, but the second is a hit because A[i] and A[i+128] are in the same cache block.
4. Calculate the cache hit rate for the line marked Line 2: **50%**
   The size of A is 8192 * 4 = 2^{15} bytes. This is exactly twice the size of our cache. At the end of Line 1, we have the second half of A inside our cache, but Line 2 starts with the first half of A. Thus, we cannot reuse any of the cache data brought in from Line 1 and must start from the beginning. Thus our hit rate is the same as Line 1 since we access memory in the same exact way as Line 1. We don’t have to consider cache hits for total, as the compiler will most likely store it in a register.

**Average Memory Access Time**

AMAT is the average (expected) time it takes for memory access. It can be calculated using this formula:

\[ \text{AMAT} = \text{hit time} + \text{miss rate} \times \text{miss penalty} \]

Miss rates can be given in terms of either local miss rates or global miss rates. The **local miss rate** of a cache is the percentage of accesses into the particular cache that miss at the cache, while the **global miss rate** is the percentage of all accesses that miss at the cache.

**Exercises**

Suppose your system consists of:

- A L1$ that hits in 2 cycles and has a local miss rate of 20%
• A L2$ that hits in 15 cycles and has a global miss rate of 5%
• Main memory hits in 100 cycles

1. What is the local miss rate of L2$?
   \[ \text{Local miss rate} = \frac{5\%}{20\%} = 0.25 = 25\% \]

2. What is the AMAT of the system?
   \[ \text{AMAT} = 2 + 20\% \times 15 + 5\% \times 100 = 10 \text{ (using global miss rates)} \]
   Alternatively, \[ \text{AMAT} = 2 + 20\% \times (15 + 25\% \times 100) = 10 \]

3. Suppose we want to reduce the AMAT of the system to 8 or lower by adding in a L3$. If the L3$ has a local miss rate of 30%, what is the largest hit time that the L3$ can have?
   Let \( H \) = hit time of the cache. Using the AMAT equation, we can write:
   \[ 2 + 20\% \times (15 + 25\% \times (H + 30\% \times 100)) \leq 8 \]
   Solving for \( H \), we find that \( H \leq 30 \). So the largest hit time is 30 cycles.

Now, what if we have a cache system with the following properties. What is the AMAT?
• L1$ hits in 1 cycle (local miss rate 25%)
• L2$ hits in 10 cycles (local miss rate 40%)
• L3$ hits in 50 cycles (global miss rate 6%)
• Main memory hits in 100 cycles (always hits)

The AMAT is \[ 1 + 0.25 \times (10 + 0.4 \times (50)) + 0.06 \times 100 = 14.5 \text{ cycles} \].
Alternatively, we can calculate the global hit rates for each hierarchy:
• L1$: 0.75
• L2$: 0.25 * 0.6 = 0.15
• L3$: 0.94 \text{ – (}0.75 + 0.15\text{)} = 0.04
• Main Memory: 1 – 0.75 – 0.15 – 0.04 = 0.06

And the following hit times:
• L1$: 1 cycle
• L2$: 1 + 10 = 11 cycles
• L3$: 1 + 10 + 50 = 61 cycles
• Main Memory: 1 + 10 + 50 + 100 = 161 cycles
Thus, our AMAT = \[ 0.75 \times 1 + 0.15 \times 11 + 0.04 \times 61 + 0.06 \times 161 = 14.5 \text{ cycles} \].

**Flynn Taxonomy**

1. Explain SISD and give an example if available.
   Single Instruction Single Data; each instruction is executed in order, acting on a single stream of data. For example, traditional computer programs.

2. Explain SIMD and give an example if available.
   Single Instruction Multiple Data; each instruction is executed in order, acting on multiple streams of data. For example, the SSE Intrinsics.

3. Explain MISD and give an example if available.
   Multiple Instruction Single Data; multiple instructions are executed simultaneously, acting on a single stream of data. There are no good modern examples.

4. Explain MIMD and give an example if available.
   Multiple Instruction Multiple Data; multiple instructions are executed simultaneously, acting on multiple streams of data. For example, map reduce or multithreaded programs.