S1:

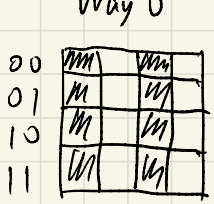
1. hit rate 是 0. 因为 block size 是 8, number of block 是 4. 所以一次访问是 32个 byte cache size 只有 32个 byte. 所以每访问一次, 都会 miss.

2. 还是 0.

3. 正常方法: step size 为 1. 可以 miss - hit - miss - hit 0.5

耗费方法: step size 为 128. rep count -> ∞ -> 1.

S2:



1. 0.75

每次读入的时候会 miss, 写入的时候 hit, 读第二个数 hit. 写入再 hit.

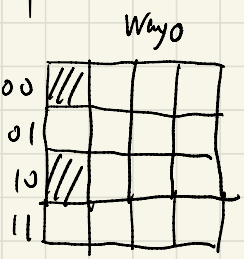
miss - hit - hit - hit - miss - hit - hit - hit - ...

所以是 0.75

2. 接近于 1. 因为 cache 的 size 跟 array 的 size 一样. 意味着在第一遍循环后理论上可以存下所有的数, 而后面的循环再读入和写时这些数据已经在 cache 中. 因而可以接近于 1.

3. cache blocking. 一次处理跟 cache size 一样大小的数据. 这样可以提升运行效率.

S3:



1. 0 ~ 0.5

理论上可以有 00 和 01 这两个 set 被使用了, 所以理论上最大到 0.5.

best case 是 1 miss - 1 hit

worst 是 always miss

2. associative 设置为 1. LRU

hit rate 是 0.5.

第一遍循环存下了数据, 此时是 miss. 第二遍都有了, 所以是 hit.

所以是 0.5