

## 1 Hamming ECC

Recall the basic structure of a Hamming code. We start out with some bitstring, and then add parity bits at the indices that are powers of two (1, 2, 8, etc.). We don't assign values to these parity bits yet. **Note that the indexing convention used for Hamming ECC is different from what you are familiar with.** In particular, the 1 index represents the MSB, and we index from left-to-right. The  $i$ th parity bit  $P\{i\}$  covers the bits in the new bitstring where the *index* of the bit under the aforementioned convention,  $j$ , has a 1 at the same position as  $i$  when represented as binary. For instance, 4 is  $0b100$  in binary. The integers  $j$  that have a 1 in the same position when represented in binary are 4, 5, 6, 7, 12, 13, etc. Therefore,  $P4$  covers the bits at indices 4, 5, 6, 7, 12, 13, etc. A visual representation of this is:

| Bit position        |     | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14  | 15  | 16  | 17  | 18  | 19  | 20  |     |
|---------------------|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|
| Encoded data bits   |     | p1 | p2 | d1 | p4 | d2 | d3 | d4 | p8 | d5 | d6 | d7 | d8 | d9 | d10 | d11 | p16 | d12 | d13 | d14 | d15 |     |
| Parity bit coverage | p1  | X  |    | X  |    | X  |    | X  |    | X  |    | X  |    | X  |     | X   |     | X   |     | X   |     |     |
|                     | p2  |    | X  | X  |    |    | X  | X  |    |    | X  | X  |    |    | X   | X   |     |     | X   | X   |     | ... |
|                     | p4  |    |    |    | X  | X  | X  | X  |    |    |    |    | X  | X  | X   | X   |     |     |     |     | X   |     |
|                     | p8  |    |    |    |    |    |    |    | X  | X  | X  | X  | X  | X  | X   | X   |     |     |     |     |     |     |
|                     | p16 |    |    |    |    |    |    |    |    |    |    |    |    |    |     |     | X   | X   | X   | X   | X   |     |

Source: [https://en.wikipedia.org/wiki/Hamming\\_code](https://en.wikipedia.org/wiki/Hamming_code)

- 1.1 How many bits do we need to add to  $0011_2$  to allow single error correction?
- 1.2 Which locations in  $0011_2$  would parity bits be included?
- 1.3 Which bits does each parity bit cover in  $0011_2$ ?
- 1.4 Write the completed coded representation for  $0011_2$  to enable single error correction. Assume that we set the parity bits so that the bits they cover have even parity.
- 1.5 How can we enable an additional double error detection on top of this?
- 1.6 Find the original bits given the following SEC Hamming Code:  $0110111_2$ . Again, assume that the parity bits are set so that the bits they cover have even parity.

- 1.7 Find the original bits given the following SEC Hamming Code: 1001000<sub>2</sub>

## 2 Amdahl's Law

In the programs we write, there are sections of code that are naturally able to be sped up. However, there are likely sections that just can't be optimized any further to maintain correctness. In the end, the overall program speedup is the number that matters, and we can determine this using Amdahl's Law:

$$\text{True Speedup} = \frac{1}{S + \frac{1-S}{P}}$$

where  $S$  is the non-sped-up part and  $P$  is the speedup factor (determined by the number of cores, threads, etc.). You run a profiling program on a different program to find out what percent of this program each function takes. You get the following

results:

| Function | % Time |
|----------|--------|
| f        | 30%    |
| g        | 10%    |
| h        | 60%    |

- (a) We don't know if these functions can actually be parallelized. However, assuming all of them can be, which one would benefit the most from parallelism?
- (b) Let's assume that we verified that your chosen function can actually be parallelized. What speedup would you get if you parallelized just this function with 8 threads?

- 2.2 You are going to run a convolutional network to classify a set of 100,000 images using a computer with 32 threads. You notice that 99% of the execution of your project code can be parallelized on these threads. What is the speedup?

## 3 Warehouse Scale Computing

Sources speculate Google has over 1 million servers. Assume each of the 1 million servers draw an average of 200W, the PUE is 1.5, and that Google pays an average of 6 cents per kilowatt-hour for datacenter electricity. Estimate Google's annual power bill for its datacenters.

- 3.2 Google reduced the PUE of a 50,000-machine datacenter from 1.5 to 1.25 without decreasing the power supplied to the servers. What's the cost savings per year?

## 4 RAID

4.1 Fill out the following table:

|        | Configuration | Pro/Good for | Con/Bad for |
|--------|---------------|--------------|-------------|
| RAID 0 |               |              |             |
| RAID 1 |               |              |             |
| RAID 2 |               |              |             |
| RAID 3 |               |              |             |
| RAID 4 |               |              |             |
| RAID 5 |               |              |             |