

17

Find $65537^{89} \bmod 70001$ three different ways:

memory-intensive binary

First, calculate: $89_{\text{DEC}} = 1011001_{\text{BIN}}$.

| n | 89 binary | 65537^n mod 70001 |
|----|--------------|------------------------|
| 1 | 1 | 65537 |
| 2 | 0 | 47012 |
| 4 | 0 | 56572 |
| 8 | 1 | 15465 |
| 16 | 1 | 42809 |
| 32 | 0 | 54302 |
| 64 | 1 | 55081 |

| | | | | | | |
|--------------|----------|-----------------|----------|------------------|----------|---------------------|
| 65537^1 | \equiv | 65537^1 | \equiv | (65537) | \equiv | $65537 \bmod 70001$ |
| 65537^9 | \equiv | 65537^{8+1} | \equiv | $(15465)(65537)$ | \equiv | $55227 \bmod 70001$ |
| 65537^{25} | \equiv | 65537^{16+9} | \equiv | $(42809)(55227)$ | \equiv | $68870 \bmod 70001$ |
| 65537^{89} | \equiv | 65537^{64+25} | \equiv | $(55081)(68870)$ | \equiv | $4279 \bmod 70001$ |

Thus, $65537^{89} \bmod 70001 \equiv 4279$.

memorylessly parsing exponent from right

First, calculate: $89_{\text{DEC}} = 1011001_{\text{BIN}}$. This will be the same as before, except that the computer will only “remember” one line of the table at a time.

| exponent | current position | operation | partial products |
|----------|------------------|----------------|------------------|
| 1 | 65537 | (65537) | 65537 |
| 01 | 47012 | | 65537 |
| 001 | 56572 | | 65537 |
| 1001 | 15465 | (15465)(65537) | 55227 |
| 11001 | 42809 | (42809)(55227) | 68870 |
| 011001 | 54302 | | 68870 |
| 1011001 | 55081 | (55081)(68870) | 4279 |

Thus, $65537^{89} \bmod 70001 \equiv 4279$.

memorylessly parsing exponent from left

First, calculate: $89_{\text{DEC}} = 1011001_{\text{BIN}}$. The computer only knows “one line at a time” in the table below.

| exponent | current position | operation | partial products |
|----------|------------------|----------------|------------------|
| 1 | 65537 | | 65537 |
| 10 | 47012 | | 47012 |
| 101 | 56572 | (56572)(65537) | 26200 |
| 1011 | 10194 | (10194)(65537) | 64635 |
| 10110 | 23545 | | 23545 |
| 101100 | 29106 | | 29106 |
| 1011001 | 7134 | (7134)(65537) | 4279 |

Thus, $65537^{89} \bmod 70001 \equiv 4279$.

K

Using the AHU approach, we will find $\gcd(51\ 195\ 673, 20\ 487\ 827)$. Our worst case estimate is $1.5 \log(20487827) \leq 37$ steps.

| | 7499 #copies 51 195 673 | 3001 #copies 20 487 827 | |
|------------|-------------------------------|-------------------------------|------|
| 51 195 673 | 1 | -0 | 7499 |
| 2 | | | 2 |
| 20 487 827 | -0 | 1 | 3001 |
| 2 | | | 2 |
| 10 220 019 | 1 | -2 | 1497 |
| 213 | | | 213 |
| 47 789 | -2 | 5 | 7 |
| 1 | | | 1 |
| 40 962 | 427 | -1067 | 6 |
| 6 | | | 6 |
| 6827 | -429 | 1072 | 1 |
| NaN | | | NaN |
| 0 | 3001 | -7499 | 0 |

Thus, the $\gcd = 6827$

$$\begin{aligned}
 &= (-429)(51195673) + (1072)(20487827) \\
 &= (-429 + 3001t)(51195673) + (1072 - 7499t)(20487827), \forall t \in \mathbb{Z}
 \end{aligned}$$

Our calculations only took 5 steps, much lower than our 37 estimate. This was due to the gcd and quotients being relatively high.

24.66

Since $(\gcd(51\ 195\ 673, 20\ 487\ 827))(\text{lcm}(51\ 195\ 673, 20\ 487\ 827)) =$
 $(51\ 195\ 673)(20\ 487\ 827) = 1\ 048\ 888\ 091\ 572\ 571,$
 $\text{lcm}(51\ 195\ 673, 20\ 487\ 827) = 153\ 638\ 214\ 673.$

e

Solve the congruence

$$257x \equiv 65537 \pmod{78125}$$

First we must solve $257^{-1} \pmod{78125}$ by computing the $\gcd(78125, 257)$ using the Euclidean Algorithm. Our worst case estimate is $1.5 \log_2(257) \leq 13$ steps.

| Remainder | Quotient | #copies 78125 | #copies 257 |
|-----------|----------|------------------|----------------|
| 78125 | | 1 | -0 |
| | 303 | | |
| 257 | | -0 | 1 |
| | 1 | | |
| 254 | | 1 | -303 |
| | 84 | | |
| 3 | | -1 | 304 |
| | 1 | | |
| 2 | | 85 | -25839 |
| | 2 | | |
| 1 | | -86 | 26143 |
| | NaN | | |
| 0 | | 257 | -78125 |

Thus, the $\gcd(78125, 257) = 1$, which divides 65537, so there exists a solution to our problem. Further, $\gcd(78125, 257) = 1 = (78125)(-86) + (257)(26143)$, meaning

$$257^{-1} \equiv 26143 \pmod{78125}.$$

Now, we compute

$$\begin{aligned} x &\equiv (257)^{-1}(257)x \equiv (257)^{-1}(65537) \\ &\equiv (26143)(65537) \\ &\equiv 52541 \pmod{78125}. \end{aligned}$$

Check: $(257)(52541) \equiv (65537) \pmod{78125}$.

g

Solve the congruence

$$67159x \equiv 3653 \pmod{140219}$$

First we must solve $67159^{-1} \pmod{140219}$ by computing the $\gcd(140219, 67159)$ using the Euclidean Algorithm.

| | | 499 #copies 140219 | 239 #copies 67159 | | |
|--------|-----|--------------------------|-------------------------|-----|-----|
| 140219 | | 1 | -0 | 499 | |
| | 2 | | | | 2 |
| 67159 | | -0 | 1 | 239 | |
| | 11 | | | | 11 |
| 5901 | | 1 | -2 | 21 | |
| | 2 | | | | 2 |
| 2248 | | -11 | 23 | 8 | |
| | 1 | | | | 1 |
| 1405 | | 23 | -48 | 5 | |
| | 1 | | | | 1 |
| 843 | | -34 | 71 | 3 | |
| | 1 | | | | 1 |
| 562 | | 57 | -119 | 2 | |
| | 2 | | | | 2 |
| 281 | | -91 | 190 | 1 | |
| | NaN | | | | NaN |
| 0 | | 239 | -499 | 0 | |

Thus, the $\gcd(78125, 257) = 281$, which divides 3653, so there exists a solution to our problem. However, since the $\gcd \neq 1$, and the solution exists, we can divide our equation through by 281 and get the following new equation:

$$239x \equiv 13 \pmod{499} \tag{1}$$

Further, $\gcd(499, 239) = 1 = (499)(-91) + (239)(190)$, meaning

$$239^{-1} \equiv 190 \pmod{499}.$$

Now, we compute

$$\begin{aligned} x &\equiv (239)^{-1}(239)x \equiv (239)^{-1}(13) \\ &\equiv (190)(13) \\ &\equiv 474 \pmod{499}. \end{aligned}$$

Equivalently $x \equiv 474 + 499t \pmod{140219} \forall t \in \mathbb{Z}$, a set of solutions.

44

Solve the congruence

$$120979x \equiv 3111 \pmod{156433}$$

First we must solve $120979^{-1} \pmod{156433}$ by computing the $\gcd(156433, 120979)$ using the Euclidean Algorithm.

| Remainder | Quotient | #copies 78125 | #copies 257 |
|-----------|----------|------------------|----------------|
| 156433 | | 1 | -0 |
| | 1 | | |
| 120979 | | -0 | 1 |
| | 3 | | |
| 35454 | | 1 | -1 |
| | 2 | | |
| 14617 | | -3 | 4 |
| | 2 | | |
| 6220 | | 7 | -9 |
| | 2 | | |
| 2177 | | -17 | 22 |
| | 1 | | |
| 1866 | | 41 | -53 |
| | 6 | | |
| 311 | | -58 | 75 |
| | NaN | | |
| 0 | | 389 | -503 |

Thus, the $\gcd(156433, 120979) = 311$, which does not divide 3111, so there does not exist a solution to our problem.

45

There was one solution to problem (e), a set of solutions to problem (g), and no solution for (44).

19.6

Find a FECC with a plain-text message space of at least 13 messages with the ability to correct at least 5 errors. These specifications require the following:

Message Hamming dimension: 4 (max 16 messages)

Minimum ball radius : 5 (to correct for five errors)

Minimum FECC dimension : 11 (since the ball centers must be far enough apart)

| n | 2^n | Ball Cardinality | Balls fitting |
|-----|--------|------------------|---------------|
| 11 | 2048 | 1024 | 2 |
| 12 | 4096 | 1586 | 2 |
| 13 | 8192 | 2380 | 3 |
| 14 | 16384 | 3473 | 4 |
| 15 | 32768 | 4944 | 6 |
| 16 | 65536 | 6885 | 9 |
| 17 | 131072 | 9402 | 13 |
| 18 | 262144 | 12616 | 20 |

Thus, our “hunting license” is issued at \mathbb{H}_{17} , with the possibility of 13 balls fitting with our design specification. However, this choice is very optimistic since we want a linear code with constraints on the basis vectors b_i . Our final choice is given below, using \mathbb{H}_{22} .

$$b_1 = 1111111111100000000000$$

$$b_2 = 0011100011111100011100$$

$$b_3 = 0000111001100110001111$$

$$b_4 = 0000011111111111100000$$

To generate the ball centers, we simply take all linear combinations of our basis vectors, resulting in 16 ball centers, shown below.

$$\begin{array}{ll}
 \text{ball}_1 = 00000000000000000000 & \text{ball}_9 = 1111111111100000000000 \\
 \text{ball}_2 = 00000111111111110000 & \text{ball}_{10} = 1111100000011111000000 \\
 \text{ball}_3 = 0000111001100110001111 & \text{ball}_{11} = 1111000110000110001111 \\
 \text{ball}_4 = 0000100110011001101111 & \text{ball}_{12} = 1111011001111001101111 \\
 \text{ball}_5 = 0011100011111100011100 & \text{ball}_{13} = 1100011100011100011100 \\
 \text{ball}_6 = 0011111100000011111100 & \text{ball}_{14} = 1100000011100011111100 \\
 \text{ball}_7 = 0011011010011010010011 & \text{ball}_{15} = 1100100101111010010011 \\
 \text{ball}_8 = 0011000101100101110011 & \text{ball}_{16} = 1100111010000101110011
 \end{array}$$

To prove that all ball centers are far enough apart, we can compute the Hamming distance between each ball center. Equivalently, we can just compute the first row of the 16×16 matrix (how far the ball center is from the zero vector) because each distance between two balls is the Hamming weight of another ball. All ball centers are the required minimum of 11 apart from each other (to allow for 5 errors).

$$\text{Distances} = [0 \ 12 \ 11 \ 11 \ 12 \ 12 \ 11 \ 11 \ 11 \ 11 \ 12 \ 16 \ 11 \ 11 \ 12 \ 12]$$

To emphasize the balls are widely spaced, a table is provide below to summarize our results. We can ignore the 0 since it is obvious a ball center is 0 distance away from itself.

| | | | |
|------------------|----|----|----|
| Hamming Distance | 11 | 12 | 16 |
| Number of Pairs | 8 | 6 | 1 |

As comparison, we will compare our results to the best pre-Shannon Hamming space. Pre-Shannon code is say it 11 times, which uses \mathbb{H}_{44} . This has an overhead of 1000%. Our code, using \mathbb{H}_{22} has an overhead of 450%.

We have done more than twice as well as the dumb approach based on overhead, which is the percentage of more bits needed to send the four bit message without any correction, i.e. 22 bits versus 44 bits. Compared to saying it once, we have almost five times the overhead. The usage of of this code corresponds to both business practice and common sense since in both we want to minimize the number of bits needed (to remain competitive) and because we have better things to do than saying something eleven times.