Kick Start 2017 - Round C

Analysis: Ambiguous Cipher

Ambiguous Cipher: Analysis

Small dataset

There are only $26^2 + 26^3 + 26^4 = 475228$ words with lengths in the [2, 4] range. The statement tells us exactly how to encrypt a word, so we can encrypt all possible words and then create a reverse dictionary. For example, when we find that `SOUP` encrypts to `OMDU`, we can create an entry in our reverse dictionary with key `OMDU` and value `SOUP`. If we ever find that a word has encrypted to a key we already have in the dictionary (for example, `ABC` and `CBA` both encrypt to `BCB`), then that key's decryption is ambiguous, and we can replace the value with `AMBIGUOUS`.

Once we have done this, solving the problem is easy; we just look up each word in our dictionary and output the value as our answer. We do not need to worry about words with no valid decryption (and thus no dictionary key), because the problem statement guarantees that these will not appear in the datasets.

This method will not work for the Large dataset, though; in that case, there are just too many possible words to stuff into a dictionary in memory! We need another approach.

Large dataset: words with even lengths

For now, let's assume our encrypted word has an even length. We will consider words of odd length in the next section.

When a word is being encrypted, every letter in the word (except for the first and last) is encrypted by summing the two letters adjacent to that letter. For the first and last letters, the encrypted value is just the unencrypted value of the one letter adjacent to it. This means that we can easily find the the second and second-to-last letters of the unencrypted word: they are just the first and last letters of the encrypted word, respectively.

Moreover, once we have these "footholds" into the decrypted word, we can use them to decrypt the rest of the encrypted word! For example, we can decrypt the encrypted word `ADMZZO` as follows. In this explanation, we will denote a letter with E or D (referring to the encrypted or decrypted words) followed by a number (giving the letter's position in the word, counting starting from 1.)

- The letter values of the encrypted word are: 0 3 12 25 25 14.
- We know that D2 and D5 equal E1 and E6, as explained above, so we know that our decrypted word has letter values _ 0 _ _ 14 _.
- We will now determine the letters in the even-numbered positions of the decrypted word, going from left to right. We will start by determining D4. We already know that D2 has value 0. Since we know that (D2 + D4) % 26 = E3, then D4 = (E3 - D2) % 26. E3 has value 12, so D4 = (12 - 0) % 26 = 12. So now our decrypted letter values are: _ 0 _ 12 14 _.
- Now that we know D4, we can find D6 in the same way: D6 = (E5 - D4) % 26 = (25 - 12) % 26 = 13. So we have _ 0 _ 12 14 13.
- Then, we do the same thing from right to left, starting with D5. We have D3 = (E4 - D5) % 26 = (25 - 14) = 11, and D1 = (E2 - D3) % 26 = (3 - 11) % 26 = 18. (It is OK to take the modulus of a negative number, but we can also always safely add 26 if we want to only

handle positive values, e.g., (3 - 11 + 26) % 26.) So we know all the letter values in the decrypted word: 18 0 11 12 14 13.
- Now we just need to convert this list of decrypted letter values back to uppercase letters, which results in the decrypted word `SALMON`.

This method will allow us to decrypt any word with an even length. Because the method was deterministic and we had no choices about how to decrypt, we can also see that any encrypted word with an even length has exactly one possible decryption; that is, in mathematical terms, decryption is both one-to-one and onto. But what about words with odd lengths?

Large dataset: words with odd lengths

Let's look back at the example `BCB` from the problem statement. We know that D2 = E1 and D2 = E3, so D2 is `B`. But what about D1 and D3? All we know is that (D1 + D3) % 26 = E2. There are 26 different pairs for which this is valid! So the decryption is ambiguous.

In fact, this is the case for *any* word of odd length that could appear in our datasets. Our method above does not apply! Knowing the second and second-to-last letters of the decrypted word does not help us get the entire word, since now both of those letters are in even-numbered positions of the decrypted word; they can only get us letters in the even-numbered positions of the decrypted word. We have no foothold that we can use to get any of the letters in the odd-numbered positions. The first letter of the decrypted word could be any letter, and any choice determines all the others. We have no way of knowing which of the 26 possibilities Calvin originally encrypted.

The statement tells us that the datasets do not include words like `APE` which have no valid decryption, so any word of odd length in our dataset must have multiple decryptions. With this knowledge, we can finish our solution: for words of even length, we use the method above, and for words of odd length, we can just report `AMBIGUOUS`. Calvin and Susie should come up with a method that can handle all words, not just words of even length!