For this TP you can start from the file `tp3_code.py` which contains the LFSR functions and Berlekamp-Massey.

Cryptanalysis of the Geffe Cipher

The Geffe cipher is a stream cipher with 3 combined registers, proposed by Geffe in 1973. It combines three binary LFSRs using the Boolean function:

$$f(x_1, x_2, x_3) = x_3 + x_2x_3 + x_1x_2 \ .$$

In this exercise, we suppose that the internal LFSRs are the following:

- LFSR1 of length 13, polynomial $P_1 = 1 + X + X^3 + X^4 + X^{13}$
- LFSR2 of length 11, polynomial $P_2 = 1 + X^2 + X^{11}$
- LFSR3 of length 9, polynomial $P_3 = 1 + X^4 + X^9$

During initialization, the three LFSRs are initialized with their respective initial states S_1, S_2, S_3. We note respectively $s_1(t), s_2(t), s_3(t)$ the output bits at time t, and $z(t)$ the output bit of the combined LFSR.

Question 1. *What is the value of $z(t)$ depending on $s_1(t), s_2(t)$ and $s_3(t)$?*

Question 2. *Program in Python a function* `geffe(S1,S2,S3,N)` *that takes as input the three initial states, and an integer N, and returns the N first bits of the sequence.*

Check that the 20 first bits of the sequence generated with the generator, initialized with the states:

$$\begin{cases} S_1 = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1] \\ S_2 = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1] \\ S_3 = [1, 0, 1, 0, 1, 0, 1, 0, 1] \end{cases}$$

are:

$$Z = 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, \ldots$$

Question 3. *What is the linear complexity of each internal LFSR? In theory, what is the linear complexity of the Geffe generator? Check with the Berlekamp-Massey algorithm.*

Correlation Attack

Question 4. *What is the complexity of exhaustive search to determine the initial state of the Geffe generator?*

The goal of this section is to describe the *correlation attack* proposed by Siegenthaler in 1985 on the Geffe generator.

We assume that at each time t, the bits $s_1(t), s_2(t), s_3(t)$ produced by the internal LFSRs are independent random variables, uniformly distributed in $\mathbb{F}_2$. Then, $z(t) = f(s_1(t), s_2(t), s_3(t))$ is also a random variable taking values in $\mathbb{F}_2$.

Question 5. *Show that:*

$$\Pr[z(t) = s_1(t)] = \Pr[z(t) = s_3(t)] = \frac{3}{4}$$

and

$$\Pr[z(t) = s_2(t)] = \frac{1}{2}$$

Question 6. *Let x be a random variable uniformly distributed over $\mathbb{F}_2$ and independent from $z(t)$. What is the probability: $p = \Pr[z(t) = x]$?*

We have shown that the output of the combined LFSR is strongly correlated with the output of the two LFSRs LFSR1 and LFSR3. The goal of a correlation attack is to exploit this property to deduce the initial states of each register.

We suppose that using a known plaintext, we have determined ℓ bits of the sequence, from $z(t_0)$ to $z(t_0 + \ell - 1)$.

Finding the internal state of LFSR1. We do an exhaustive search on the internal state of LFSR1 at time t_0, and for each candidate state $\bar{S}_1$, we compute the sequence $s_1(t)$. When the internal state is correct, we expect that $(s_1(t))$ coïncides with $\simeq 3/4$ of the values if ℓ is large enough. Otherwise, we expect that only $\ell/2$ elements coincide.

Finding the internal state of LFSR3. We do the same with LFSR3, and we obtain a candidate $\bar{S}_3$ for its internal state at time t_0.

Finding the internal state of LFSR2. Finally we perform an exhaustive search for the internal state of LFSR2. If the state is correct we expect to obtain exactly the output sequence.

Question 7. *What is the complexity of the attack to find S_1 and S_3? Of the whole attack? Is it better than exhaustive search?*

Question 8. *Implement this attack. The file* `tp3_code.py` *contains a sequence of 100 bits generated by the Geffe generator (also given below). Find the corresponding initial state of the three LFSRs.*

Challenge sequence:

```
challenge = [0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0,
0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0,
1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 1, 0]
```