LFSRs y cifradores de flujo

Moreno Ramírez, Eliú[1]

Instituto Nacional de Astrofísica Óptica y Electrónica, Puebla; México
eliu.moreno@inaoep.mx

El primer polinomio asociado es $f(x) = x^{32} + x^7 + x^6 + x^2 + 1$ el cual tiene grado 32 que a su vez es la longitud del registro, la secuencia tab o función de retroalimentación es $S_1(t) = S_{32} \bigoplus S_7 \bigoplus S_6 \bigoplus S_2$, y en este caso el periodo máximo del polinomio $2^n - 1 = 4294967296 - 1 = 4294967295$.

El segundo polinomio asociado es $f(x) = x^{57}+x^7+1$ el cual tiene grado 57 que a su vez es la longitud del registro, la secuencia tab o función de retroalimentación es $S_1(t) = S_{57} \bigoplus S_7$, y en este caso el periodo máximo del polinomio $2^n - 1 = 1{,}4411519e + 17 - 1$.

Y el tercer polinomio asociado es $f(x) = x^{130} + x^3 + 1$ el cual tiene grado 130 que a su vez es la longitud del registro, la secuencia tab o función de retroalimentación es $S_1(t) = S_{130} \bigoplus S_3$, y en este caso el periodo máximo del polinomio $2^n - 1 = 1{,}3611295e + 39 - 1$.

Con estos polinomios que son (32,7,6,2,0), (57,7,0), (130,3,0) que son los tres registros usados para alimentar el generador de Geffe, para implementar este se utilizó python, y para generar las semillas de cada generador se uso una función random. En este caso $n_1 = 57, n_2 = 32, n_3 = 130$ por lo que la complejidad es $(n_1 + 1)n_2 + n_1n_3 = (57 + 1) * 32 + 57 * 130 = 9266$. Una vez implementado Geffe se agregó que para encriptar el texto, primeramente se mete dicho texto en una función que convierte este en binario para realizar el xor entre la llave y este texto. Para descifrar el mensaje pasa por una función primeramente pasa por un convertidor del texto cifrado para tenerlo en forma de cadena binaria, realizar un xor entre esta y la llave utilizada para cifrar.

Referencias

1. Fernández-Conde, J.; Cuenca-Jiménez, P.; Cañas, J.M. Hybrid Training Strategies: Improving Performance of Temporal Difference Learning in Board Games. Appl. Sci. 2022, 12, 2854. https://doi.org/10.3390/app12062854
2. Poliansky, R.; Sipper, M.; Elyasaf, A. From Requirements to Source Code: Evolution of Behavioral Programs. Appl. Sci. 2022, 12, 1587. https://doi.org/10.3390/app12031587