Combining LFSRs to Achieve Nonlinearlity

D. R. Stinson

In a synchronous stream cipher defined over $\mathbb{Z}_2$, a binary keystream is generated from the key (independent of the plaintext) and then exclusive-ored (or added modulo 2) with the plaintext. There exist linear feedback shift registers that can be used to generate a keystream with period $2^m - 1$ from a "seed" of m bits. However, the linearity of the keystream makes it very vulnerable to attack. Thus one objective in designing a stream cipher is to produce a keystream that is non-linear.

Along this line, many proposals have been made which "combine" two or more LFSRs in some non-linear way. We will discuss briefly a first step in this direction, taken from the book *Cipher Systems* by Beker and Piper. (Note, however, that the methods described are too simple to yield a secure cipher.)

Perhaps the simplest approach would be to take two LFSRs and combine them by multiplying their outputs (this is the same thing as taking the logical "and" of the outputs). More precisely, this situation is described as follows. Suppose that L_1 is an m-stage LFSR with (maximal) period $2^m - 1$, and L_2 is an n-stage LFSR with (maximal) period $2^n - 1$. Further suppose that $\gcd(m, n) = 1$. L_1 is initialized with a seed $j_1, \ldots j_m$, producing a sequence $j_1, \ldots$; and L_2 is initialized with a seed $k_1, \ldots k_n$, producing a sequence $k_1, \ldots$. Then the keystream is defined to be $z_1, \ldots$, where $z_i = j_i k_i$, $i \geq 1$.

It is possible to prove that the period of the resulting keystream is $(2^m - 1)(2^n - 1)$. This is almost as long as the maximal length of a keystream produced from an $(m + n)$-stage LFSR. But one serious weakness of the keystream is that it is not *balanced*: approximately 75% of the bits produced are zeroes. This happens because the only time that a value $z_i = 1$ is when $k_i = j_i = 1$, which happens only about 25% of the time.

A fairly simple way to produce a more balanced keystream from two LFSRs is to use a device known as a *J-K flip-flop*. A *J-K* flip-flop has two binary inputs, which we denote as J and K. The output of the *J-K* flip-flop at time t will be denoted as q_t, where $q_0 = 0$. For $t \geq 1$, the value q_t is

defined recursively as follows:

$$q_t = J + q_{t-1}(1 + J + K) \bmod 2.$$

In other words, $q_t = J$ if $J \neq K$, and $q_t = J + q_{t-1} \bmod 2$ if $J = K$.

The keystream will consist of the values $q_1, \ldots$, and can be shown to have period $(2^m - 1)(2^n - 1)$. Further, the keystream is much closer to being balanced than with the previous method of multiplying the two sequences.

As an illustrative example, let us suppose that L_1 produces the sequence

$$1001011\ldots$$

having period 7; and L_2 produces the sequence

$$1000010011010111\ldots$$

having period 15. Then the beginning of the resulting keystream produced by a J-K flip-flop will be as indicated in the following table:

t	J	K	q_t
0			0
1	1	1	1
2	0	0	1
3	0	0	1
4	1	0	1
5	0	1	0
6	1	0	1
7	1	0	1
8	1	1	0
9	0	1	0
10	0	0	0
11	1	1	1
12	0	0	1
13	1	1	0
14	1	1	1
15	1	1	0
16	0	1	0
17	0	0	0
18	1	0	1
19	0	0	1
20	1	1	0