Uso de los autómatas celulares de Wolfram en los criptosistemas de cifrado en flujo[1]

R. Díaz Len[2], L. Hernández Encinas[3], A. Hernández Encinas, S. Hoya White, A. Martín del Rey, G. Rodríguez Sánchez e I. Visus Ruiz[4]

Resumen

En el presente trabajo se detalla una de las líneas de investigación que desarrolla en la actualidad el Grupo de Investigación en Autómatas Celulares y sus Aplicaciones (GIACA, Universidad de Salamanca-CSIC) con relación al uso de los autómatas celulares lineales en la implementación de los criptosistemas de cifrado en flujo. Así, en el presente artículo se estudian las propiedades de los denominados autómatas celulares de Wolfram como generadores pseudoaleatorios de secuencias de bits, indicándose de forma explícita aquellos que poseen buenas propiedades pseudoaleatorias.

1 Introducción

La generación de números y secuencias de bits aleatorias o pseudoaleatorias es esencial en el desarrollo de muchos campos de la Ciencia y la Tecnología: uso del método de Monte-Carlo, aplicaciones a los juegos de ordenador, a la dinámica Browniana, a la optimización estocástica, etc. También la seguridad de gran cantidad de sistemas criptográficos depende de la generación de magnitudes impredecibles. Por ejemplo, la secuencia cifrante en el cifrado en flujo, los números primos p, q, usados en el criptosistema de clave pública RSA, la clave secreta del algoritmo de cifrado del criptosistema simétrico DES, ciertos valores en los esquemas de firma digital o la clave privada en el protocolo de firma digital DSA, etc. ([12]). En todos estos casos, las cantidades generadas deben tener un tamaño lo suficientemente grande y deben ser "aleatorias", en el sentido de que la probabilidad de poder predecir un valor sea lo suficientemente pequeña como para evitar que un adversario pueda implementar un sistema de criptoanálisis basándose en dicha probabilidad.

Las secuencias de números o de bits utilizadas en las aplicaciones mencionadas se denominan aleatorias o pseudoaleatorias, en función del procedimiento que se siga para determinarlas. Para las sucesiones aleatorias se suelen utilizar procedimientos basados en fenómenos físicos, que son difícilmente repetibles, lo que supone la imposibilidad de volver a generar exactamente la misma secuencia. Para las secuencias pseudoaleatorias se emplean algoritmos determinísticos, que permiten obtener la misma secuencia cada vez que se utiliza el mismo algoritmo con los mismos parámetros. Utilizar uno u otro método depende de la aplicación a la que estén destinados los números generados. Así por ejemplo, hay determinadas aplicaciones, como la criptografía y más concretamente en los criptosistemas de cifrado en flujo (véase, por ejemplo, [1]), en los que resulta totalmente imprescindible que los números (o secuencia de bits) generados puedan volver a repetirse de la misma forma. Este hecho, aunque merma el nivel de "azar" en el proceso de generación —ya que se puede volver a generar la misma secuencia tantas veces como se desee—, no le hace perder su carácter aleatorio, en el sentido de que conocidos $n-1$ elementos de la secuencia, la probabilidad de determinar el n-ésimo elemento debe ser menor que $1/2$. Para garantizar este hecho, es fundamental que los generadores pseudoaleatorios no sean predecibles, por lo que se basan en algoritmos que implementan problemas

[1]Trabajo parcialmente subvencionado por la Fundación "Samuel Solórzano Barruso".

[2]Dpto. Matemática Aplicada. E.P.S. de Ávila. Universidad de Salamanca. E-mail: rauldlen@usal.es

[3]Dpto. Tratamiento de la Información y Codificación. Instituto de Física Aplicada, C.S.I.C. E-mail: luis@iec.csic.es

[4]Dpto. Matemática Aplicada. E.T.S.I.I. de Béjar. Universidad de Salamanca. E-mails: {ascen, sarahw, delrey, gerardo, ivisus}@usal.es

matemáticos difíciles de resolver (computacionalmente hablando).

Por otra parte, los sistemas dinámicos han sido propuestos para su uso en determinadas aplicaciones criptográficas (para un estudio de tales aplicaciones puede verse [15]). En particular, la propuesta de la utilización de los autómatas celulares con propósitos criptográficos es relativamente reciente. Como es bien sabido, un autómata celular lineal es un sistema dinámico en el que el tiempo y el espacio son discretos. Estos autómatas están formados por un conjunto de entes idénticos denominados células, cada uno de los cuales puede estar en uno de un número finito de posibles estados, y tales que se van actualizando de manera sincronizada en pasos discretos del tiempo según una determinada regla. De esta manera el estado de toda célula en un determinado instante del tiempo depende de los estados en el instante anterior de un conjunto de células denominado vecindad de la primera. Esta regla, que rige la evolución de los estados de las células del autómata, es esencialmente una máquina de estados finitos, usualmente especificada como una tabla de reglas, denominada función de transición, con una entrada para cada una de las posibles configuraciones de las distintas vecindades. En 1986, Stephen Wolfram ([20]) propuso el uso del autómata celular lineal con número de Wolfram 30 para generar secuencias pseudoaleatorias para la implementación de un método de cifrado en flujo. Para analizar sus propiedades aleatorias, se implementaron siete tests estadísticos, obteniéndose como resultado que dicho generador era muy superior a los generadores de números aleatorios denominados registros de desplazamiento realimentados linealmente (LFSR). Una implementación en software y hardware de este modelo se puede encontrar en [18]. No obstante, hay que indicar que Meier y Staffelbach demostraron en 1991 que esta propuesta es vulnerable en determinados supuestos ([11]). Hortensius et al. describieron el uso del anterior autómata celular como generador de números pseudoaleatorios en una implementación en VLSI de una computadora bidimensional Ising ([8, 9]). Los propiedades de pseudoaleatoriedad de autómatas híbridos (se denomina así a aquel autómata cuya regla de transición no es única sino que varía en función de la posición de la célula considerada) también han sido estudiadas. El autómata híbrido más empleado está constituido por los autómatas de Wolfram con números 90 y 150 (véanse, por ejemplo, [4, 5, 6, 14, 16]). Más recientemente, en 1999, Tomassini, Sipper, Zolla y Perrenoud ([17]) introdujeron el uso de la batería de tests desarrollada por Marsaglia, DIEHARD, para estudiar el comportamiento de los autómatas celulares de Wolfram con números 30, 90, 105, 150 y 165. Sin embargo, estos estudios se limitan a la generación y análisis de las propiedades de pseudoaleatoriedad de los autómatas considerados y no tienen en cuenta los aspectos de seguridad que se requieren en criptografía.

En el presente trabajo se estudia el uso de los autómatas celulares lineales de Wolfram ([19]) como generadores pseudoaleatorios, desde el punto de vista de su posible aplicación a los cifrados en flujo (véase también [7]). Para ello se ha analizado el comportamiento de los mencionados autómatas frente a los siguientes cinco tests estadísticos ([12, §5.4.4]): test de frecuencias, test de series, test de póker, test de rachas y test de autocorrelación.

El resto del presente artículo se organiza de la siguiente forma: En la §2 se presentan las principales propiedades de los generadores de números pseudoaleatorios, así como las características de los cinco tests estadísticos que se han implementado y que se han mencionado más arriba. Los autómatas celulares, y en particular, los de Wolfram y sus propiedades, se presentan de forma más detallada en §3. En §4 se describe la forma en la que tales autómatas proporcionan secuencias pseudoaleatorias de bits, así como los resultados que se han obtenido en el presente estudio. Finalmente, las conclusiones de este trabajo aparecen en §5.

2 Generadores pseudoaleatorios y tests estadísticos

Un *generador de bits aleatorio* es un mecanismo o algoritmo que devuelve como salida una secuencia de dígitos binarios estadísticamente independientes. Dichos generadores se pueden diseñar de dos maneras principalmente: mediante hardware y mediante software. Los diseñados mediante *hardware* se basan en la aleatoriedad de algunos fenómenos físicos, como la emisión de partículas durante un proceso radiactivo entre dos instantes de tiempo,

el ruido producido por un diodo semiconductor en una corriente, la frecuencia de inestabilidad de un oscilador libre, etc. Por su parte, los generadores implementados via *software* son más difíciles de construir. Algunos de los procesos más utilizados son: los sistemas de reloj de un ordenador, el tiempo transcurrido entre pulsaciones o movimientos del ratón, etc. Como ya se ha comentado, estos generadores no son determinísticos, es decir, dada una misma entrada para un mismo método, no producen la misma secuencia aleatoria de salida. De un modo ideal, la privacidad requerida en los algoritmos y protocolos criptográficos, debería llevarse a cabo mediante generadores de bits aleatorios. Sin embargo, la generación de bits puramente aleatorios no es un procedimiento muy eficaz ni muy fácil de implementar. Por esta razón, estos generadores se suelen sustituir por los denominados generadores de bits pseudoaleatorios ([2, 10, 13]). Éstos son algoritmos que ante una misma entrada siempre producen una misma salida, esto es, un *generador de bits pseudoaleatorio* es un algoritmo determinístico al que proporcionándole una secuencia binaria realmente aleatoria de longitud k, produce una secuencia binaria de longitud $l \gg k$ que "parece" ser aleatoria. En definitiva, se trata de generar secuencias que parezcan aleatorias de tal forma que el tiempo que se tarde en distinguir dicha salida de una secuencia de bits verdaderamente aleatoria sea tan grande que haga inviable dicho procedimiento.

La entrada (de longitud k) de un generador de bits pseudoaleatorio se llama *semilla* y la salida (de longitud l) recibe el nombre de *secuencia pseudoaleatoria*. De hecho, el número de posibles secuencias pseudoaleatorias de salida es una pequeña fracción del orden de $2^k/2^l$ de todas las posibles secuencias binarias de longitud l. Para tener alguna garantía de que las secuencias generadas no son previsibles y por lo tanto son fiables, se implementan una serie de tests estadísticos diseñados para estudiar ciertas propiedades que un generador aleatorio debe tener. Además, si estas secuencias pseudoaleatorias se pretenden usar para fines criptográficos es necesario que satisfagan unos requisitos mínimos de seguridad, entre los que cabe destacar el siguiente: la longitud k de la secuencia aleatoria de entrada debe ser lo suficientemente grande como para que una búsqueda exhaustiva (criptoanálisis por fuerza bruta) realizada sobre las 2^k posibles secuencias requiera un tiempo de ejecución inviable. En resumen, dos de los requerimientos que deben tener las secuencias producidas por generadores pseudoaleatorios para que sean estadísticamente indistinguibles (en un tiempo de cómputo razonable) de verdaderas secuencias aleatorias son: deben tener buenas propiedades aleatorias (lo cual queda demostrado si pasan los diferentes tests estadísticos) y la secuencia de entrada debe tener una longitud suficiente como para asegurar buenas propiedades criptográficas. Así pues, para que un generador sea utilizado en la implementación de criptosistemas de cifrado en flujo, es condición necesaria (aunque no suficiente) que pase una serie de tests estadísticos, que han sido diseñados *ad hoc* y que están basados en los postulados de Golomb ([3]). No obstante, antes de enunciarlos se verán algunas definiciones.

Una secuencia infinita de bits $s = (s_0, s_1, \dots)$ se dice que es *n-periódica* si $s_i = s_{i+n}$, para todo $i \geq 0$ y se denominará *ciclo* de s a la subsecuencia $(s_0, \dots, s_{n-1})$. El *periodo* de una secuencia periódica es el menor entero positivo n para el que s es n-periódica. Por otra parte, una *racha* de longitud k es una subsecuencia de s de k bits iguales entre dos bits distintos. Si los bits iguales son ceros se denomina *hueco* (gap), y si son unos se denota por *bloque* (block). Finalmente, la *función de autocorrelación* de s se define como sigue:

$$C(t) = \frac{1}{n}\sum_{i=0}^{n-1}(2\,s_i - 1)\,(2\,s_{i+t} - 1),\ 0 \leq t \leq n-1,$$

y mide las coincidencias existentes entre la secuencia s y ella misma desplazada cíclicamente t posiciones. Si s es una secuencia aleatoria de periodo n, el producto $|n \cdot C(t)|$ debe ser muy pequeño para todos los valores de t, tales que $0 \leq t \leq n$. Si s es una secuencia n-periódica, los postulados de pseudoaleatoriedad de Golomb son: *Postulado 1*: En un ciclo de s, el número de unos es aproximadamente igual al número de ceros. La diferencia entre uno y otro no debe exceder de la unidad. *Postulado 2*: En cada ciclo, la mitad de las rachas del número total de rachas observadas tiene de longitud 1, una cuarta parte longitud 2, una octava parte longitud 3, etc. Para cada una de estas longitudes debe haber el mismo número de bloques que de huecos, o como mucho la diferencia entre uno y otro no debe exceder de la unidad. *Postulado 3*: La función de autocorrelación $C(t)$ toma dos valores: $C(0) = n$

y $C(t) = k$, con $1 \leq t \leq n-1$. Así pues, los tests de pseudoaleatoriedad (véase [12]) que se implementarán en el presente trabajo y que recogen los postulados de Golomb son los siguientes:

Test de Frecuencias (Monobit test). Este test determina si una secuencia $s = (s_0, \ldots, s_{n-1})$ posee el mismo número de ceros, n_0, que de unos, n_1, lo cual debe ser cierto en cualquier secuencia aleatoria (satisfaciendo así el Postulado 1 de Golomb). El estadístico utilizado en este test (que sigue una distribución χ^2 con 1 grado de libertad si $n \geq 10$) es:

$$X_f = \frac{(n_0 - n_1)^2}{n}.$$

Test de Series (Two-bit test). Este test determina si el número de veces que aparecen los grupos de bits 00, 01, 10 y 11 en s, denotados por n_{00}, n_{01}, n_{10} y n_{11}, respectivamente, sea el mismo, como cabe esperarse en cualquier secuencia aleatoria. El estadístico empleado en este test (que sigue una distribución χ^2 con 2 grados de libertad si $n \geq 21$) es:

$$X_s = \frac{4\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right)}{n-1} - \frac{2\left(n_0^2 + n_1^2\right)}{n} + 1.$$

Test de Póker. Sea m un entero positivo tal que $\lfloor \frac{n}{m} \rfloor \geq 5 \cdot (2^m)$ y sea $k = \frac{n}{m}$. Se divide la secuencia s en k partes disjuntas, cada una de ellas de longitud m. Sea n_i el número de sucesos de tipo i de cada una de las secuencias de longitud m, tal que $1 \leq i \leq 2^m$. El test determina si cada posible secuencia de longitud m aparece el mismo número de veces en s. El estadístico de este test (que sigue una distribución χ^2 con $2^m - 1$ grados de libertad) es:

$$X_p = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k.$$

Test de Rachas. Este test determina si el número de rachas de distintas longitudes en s tiene un valor adecuado para una secuencia aleatoria. El número de rachas (huecos o bloques) de longitud i en una secuencia aleatoria de longitud n es $e_i = (n - i + 3)/2^{i+2}$. Sea k el mayor valor entero de i tal que $e_i \geq 5$ y sean H_i y B_i el número de huecos y bloques respectivamente, de longitud i tal que $1 \leq i \leq k$. El estadístico que se emplea en este test (que sigue una distribución χ^2 con $2k-2$ grados de libertad) es el siguiente:

$$X_r = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(H_i - e_i)^2}{e_i}.$$

Test de Autocorrelación. Este test comprueba las correlaciones existentes entre la secuencia s y variaciones no cíclicas de sí misma. Sea d un entero tal que $1 \leq d \leq \lfloor \frac{n}{2} \rfloor$, el número de bits de s que resultan diferentes en las d variaciones realizadas es

$$A(d) = \sum_{i=0}^{n-d-1} s_i \oplus s_{i+d}.$$

El estadístico utilizado en este test (que sigue una distribución $N(0,1)$ si $n - d \geq 10$) es:

$$X_a = 2\frac{A(d) - \frac{n-d}{2}}{\sqrt{n-d}}.$$

3 Autómatas celulares

En esta sección se presentarán las principales propiedades y características de los autómatas celulares –AC–, prestando especial atención a los autómatas celulares de Wolfram. Se denomina *autómata celular lineal* a una colección finita de entes idénticos denominados *células*, dispuestas uniformemente una a continuación de otra, a modo de cadena, y susceptibles de estar en un estado determinado e ir cambiándolo con el paso discreto del tiempo según una determinada regla, en la que intervienen como variables los estados de las células vecinas. De manera más rigurosa un autómata celular lineal (finito) es una cuádrupla $A = (I, S, V, f)$, donde I es el denominado *espacio celular*, S es el *conjunto* (finito) *de estados*, V es la *vecindad* de cada célula y f es la *función de transición local*. De forma más detallada, cada una de las componentes de un autómata celular (finito) lineal se caracteriza como sigue:

El espacio celular I: La disposición espacial de las células así como su número determinan el espacio celular. Si el número de células del espacio celular es n, serán llamados *n-lineales*. Cada una de las células de un AC n-lineal se denotará por $\langle i \rangle$, donde $0 \leq i \leq n-1$, de tal forma que su espacio celular será

$\langle 0 \rangle$	$\langle 1 \rangle$	...	$\langle n-2 \rangle$	$\langle n-1 \rangle$

El conjunto de estados S: Como ya se ha comentado anteriormente, en este trabajo se supondrá que el número de estados es finito, $|S| = k$, con lo que se tomará $S = \mathbb{Z}_k$. Si A es n-lineal, entonces se denota por $a_i^{(t)} \in S$ al estado en el que se encuentra la célula $\langle i \rangle$ en el instante t. Además se denominará *configuración en el instante* t al siguiente vector

$$C^{(t)} = \left(a_0^{(t)}, \ldots, a_{n-1}^{(t)} \right) \in S \times \overset{n}{\ldots} \times S,$$

de tal forma que para $t = 0$, $C^{(0)}$ será la *configuración inicial*. Además, se denotará por $\mathcal{C}$ al conjunto de todas las posibles configuraciones del autómata celular. Obsérvese que en el caso presente se tiene que $|\mathcal{C}| = k^n$.

La vecindad V: Es el conjunto de células cuyos estados en el instante t influyen en el estado de la célula considerada en el instante $t+1$. En este trabajo y para el caso particular de los autómatas celulares de dimensión 1, se considerarán las denominadas *vecindades simétricas de radio* r; así para cada célula $\langle i \rangle$, se tiene

$$V_r = \{ \langle i - r \rangle, \ldots, \langle i \rangle, \ldots, \langle i + r \rangle \}.$$

La función de transición local f: Mediante esta función se rige la evolución de los estados de las células teniendo en cuenta la vecindad de las mismas. Así, si A es un autómata celular n-lineal en el que la vecindad es simétrica de radio r, entonces

$$a_i^{(t+1)} = f\left(a_{i-r}^{(t)}, \ldots, a_i^{(t)}, \ldots, a_{i+r}^{(t)} \right).$$

Obsérvese que, por tanto, $f\colon S^{2r+1} \to S$. Es fácil ver que si $|S| = k$, entonces el número de posibles funciones de transición que se pueden definir es $k^{k^{2r+1}}$. Además, para que dicha función de transición esté bien definida sobre todas las células del espacio celular, es necesario fijar una serie de condiciones de contorno. Las más usuales son las denominadas *condiciones de contorno periódicas*, en las que a la izquierda de la célula $\langle 0 \rangle$ se consideran las células $\langle n-1 \rangle$, $\langle n-2 \rangle$, $\langle n-3 \rangle$,... por este orden; mientras que a la derecha de $\langle n-1 \rangle$ se suponen $\langle 0 \rangle$, $\langle 1 \rangle$, $\langle 2 \rangle$,... de tal manera que el espacio celular se puede modelizar como un cilindro.

La teoría moderna de los autómatas celulares se funda sobre los cimientos establecidos por Wolfram a mediados de la década de los 80 (véanse por ejemplo [19, 20]). Dicho autor centra su atención fundamentalmente en un tipo muy particular de autómatas celulares lineales, los definidos por $A = (I, S, V, f)$, de modo que el conjunto de estados es $S = \mathbb{Z}_2 = \{0, 1\}$, las vecindades son simétricas de radio $r = 1$ y las condiciones de contorno son periódicas. Consecuentemente su función de transición será de la siguiente forma:

$$a_i^{(t+1)} = f\left(a_{i-1}^{(t)}, a_i^{(t)}, a_{i+1}^{(t)} \right).$$

Estos autómatas se denominan *autómatas celulares de Wolfram* (ACW para abreviar). Dado que existen $2^{2^3} = 256$ diferentes posibilidades de asignación para la regla de transición $f\colon S^3 \to S$, a saber:

$$f_0 = f(0,0,0),\ f_1 = f(0,0,1),\ f_2 = f(0,1,0),$$
$$f_3 = f(0,1,1),\ f_4 = f(1,0,0),\ f_5 = f(1,0,1),$$
$$f_6 = f(1,1,0),\ f_7 = f(1,1,1).$$

es posible asignar a cada ACW un número, m, con $0 \leq m \leq 255$, denominado *número de Wolfram*. Éste viene dado por el siguiente número entero decimal:

$$m = \sum_{i=0}^{7} 2^i f_i,$$

de tal forma que el ACW con número m lo denotaremos por W_m

4 Generación de secuencias de bits y resultados

Cada uno de los ACW puede considerarse como un mecanismo generador de bits. Para ello basta con considerar, por ejemplo, una configuración inicial de n células para un ACW dado e iterarla varias veces. Dado que los estados de las células son bits, si se consideran las diferentes configuraciones obtenidas, $C^{(i)} = (a_0^{(i)}, \ldots, a_{n-1}^{(i)})$, $i \geq 0$, y se concatenan los estados de las células obtenidas después de k iteraciones, se obtiene una sucesión de $k \cdot n$ bits:

$$(a_0^{(1)}, \ldots, a_{n-1}^{(1)}, a_0^{(2)}, \ldots, a_{n-1}^{(2)}, \ldots, a_0^{(k)}, \ldots, a_{n-1}^{(k)}).$$

No obstante, aquí se considerará otra selección de bits, dado que la configuración anterior proporciona toda la evolución del ACW y ello lo hace muy vulnerable, por lo que puede ser atacada fácilmente.

En este caso se considera, al igual que antes, un ACW n-lineal, con n impar, dado por la regla de transición

$$a_i^{(t+1)} = f\left(a_{i-1}^{(t)}, a_i^{(t)}, a_{i+1}^{(t)}\right),$$

y con la configuración inicial

$$C^{(0)} = \left(a_0^{(0)}, a_1^{(0)}, \ldots, a_{n-1}^{(0)}\right).$$

Iterando $k-1$ veces este ACW se obtienen k configuraciones:

$$C^{(i)} = \left(a_0^{(i)}, a_1^{(i)}, \ldots, a_{n-1}^{(i)}\right), \qquad 0 \leq i \leq k-1.$$

Se considera entonces como secuencia de bits la generada por la evolución temporal de la célula central, $\langle (n-1)/2 \rangle$, esto es:

$$s = \left(a_{\frac{n-1}{2}}^{(0)}, a_{\frac{n-1}{2}}^{(1)}, \ldots, a_{\frac{n-1}{2}}^{(k-1)}\right).$$

Así pues, sin más que elegir de forma aleatoria una determinada configuración inicial, que hace el papel de *clave o semilla* e iterar el AC de Wolfram considerado un número suficientemente grande de veces se obtendrá una secuencia de bits, de la cual se estudiarán sus propiedades de aleatoriedad. Como ya se ha comentado anteriormente, para poder determinar el comportamiento pseudoaleatorio de los distintos autómatas celulares de Wolfram, se han pasado los cinco tests estadísticos descritos anteriormente (ver §2) a 100 secuencias para un número variable de bits y para diferentes longitudes de las configuraciones iniciales de cada uno de los 256 ACW. El hecho de considerar diferentes valores para n se debe a que el criptoanálisis al texto claro conocido desarrollado en [11], es efectivo si el número de células de la configuración inicial no se elige adecuadamente. En el presente estudio se ha considerado en primer lugar el valor $n = 300$, para llevar a cabo una primera criba de entre todos los ACW, de modo que el tiempo necesario para generar las secuencias de bits no fuera excesivamente largo. Posteriormente se ha incrementado el valor del número de células a $n = 500$ de modo que quedara garantizada la seguridad de los ACW restantes, manteniendo los tiempos de computación en unos márgenes aceptables. Las cinco cribas realizadas tienen las siguientes características:

Primera Criba. La longitud de cada una de las 100 secuencias analizadas ha sido de 1000 bits, es decir, se han llevado a cabo 1000 iteraciones, siendo la configuración inicial del ACW generador de 300 células (al igual que el resto de las configuraciones). El nivel de significación de los tests considerados es 0.05 y los valores de los parámetros m del test de poker y d del test de autocorrelación se han seleccionado de forma aleatoria de entre todos los posibles, según las condiciones fijadas en dichos tests. *Segunda Criba.* Es análoga a la anterior con la salvedad de que la longitud de cada una de las 100 secuencias analizadas para cada ACW ha sido de 2500 bits y la longitud de cada una de las configuraciones de cada ACW ha sido de 500 células. *Tercera Criba.* Es análoga a las dos anteriores con la diferencia de que la longitud de cada una de las secuencias analizadas ha sido de 5000 bits. *Cuarta Criba.* También se ha hecho de forma similar a las tres anteriores, pero en esta ocasión la longitud de cada una de las secuencias analizadas ha sido de 10000 bits. *Quinta Criba.* En este caso, la longitud de cada una de las secuencias analizadas ha sido de 10000 bits, el nivel de significación de los tests considerados es 0.05. El parámetro m del test de poker ha tomado todos los posibles valores según las condiciones fijadas en dichos tests, es decir, $1 \leq m \leq 7$, y el parámetro d del test de autocorrelación ha tomado tres valores significativos entre todos los posibles valores en función de las condiciones fijadas en dichos tests, esto es $d = 2$, 2500 y 4975.

En cada una de las distintas cribas se pasan los tests a todas las secuencias de forma ordenada y eliminatoria, de manera que un ACW que tenga un porcentaje de error superior al 20% en cualquiera de los tests, no será sometido al resto de los tests. Es de destacar el hecho de que los ACW que no pasan los diferentes tests tienen porcentajes de error muy elevados, en torno al 100%, mientras que los ACW que pasan los tests lo hacen con porcentajes de error inferiores al 12%. De esta forma los resultados obtenidos al pasar los cinco tests en cada una de las cinco cribas son los siguientes: La primera criba sólo fue pasada por 24 de los 256 ACW iniciales: W_{30}, W_{45}, W_{60}, W_{75}, W_{86}, W_{89}, W_{90}, W_{101}, W_{102}, W_{105}, W_{106}, W_{120}, W_{122}, W_{126}, W_{129}, W_{135}, W_{149}, W_{150}, W_{153}, W_{161}, W_{165}, W_{169}, W_{195} y W_{225}. De los anteriores autómatas, el único que no pasó la segunda criba fue el W_{126}. La tercera criba fue pasada por todos los autómatas celulares evaluados menos uno de ellos, el W_{129}, por lo

que los AC que pasaron a una cuarta criba fueron 22. Solamente uno de los ACW evaluados, el W_{122}, fue rechazado por no pasar los tests en la cuarta criba. Finalmente y ya en la quinta criba, hay que destacar que los valores de los parámetros no influyen de forma significativa en los porcentajes de error, y todos los ACW que pasaron la cuarta criba pasaron también la quinta.

En la siguiente tabla se muestran los resultados finales obtenidos en el presente estudio. En la misma, para cada ACW definido por su número de Wolfram, se presenta el porcentaje de secuencias que no han pasado cada uno de los tests. Los tests se denotan por el estadístico que los define, según se señaló en §2.

ACW	% X_f	% X_s	% X_p	% X_r	% X_a
W_{30}	7	3	6	3	5
W_{45}	2	6	3	12	9
W_{60}	3	6	6	6	5
W_{75}	9	5	5	3	4
W_{86}	5	5	4	6	8
W_{89}	5	4	6	4	2
W_{90}	4	2	6	6	4
W_{101}	5	5	3	7	8
W_{102}	4	5	8	4	6
W_{105}	7	9	7	4	2
W_{106}	5	3	6	8	3
W_{120}	3	3	3	2	6
W_{135}	8	5	6	4	3
W_{149}	2	5	2	1	6
W_{150}	6	14	6	11	8
W_{153}	6	5	8	10	7
W_{161}	10	6	10	4	8
W_{165}	4	4	4	7	7
W_{169}	6	6	6	10	4
W_{195}	5	7	8	4	2
W_{225}	4	6	5	10	5

Consecuentemente los 21 ACW con buenas propiedades pseudoaleatorias para su aplicación en los criptosistemas de cifrado en flujo son los siguientes:

$W_{30}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_i^{(t)} + a_{i+1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{45}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_{i+1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{60}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_i^{(t)} \pmod 2$,

$W_{75}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{86}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_i^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{89}: a_i^{(t+1)} = 1 + a_i^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{90}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{101}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{102}: a_i^{(t+1)} = a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{105}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{106}: a_i^{(t+1)} = a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{120}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{135}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{149}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{150}: a_i^{(t+1)} = a_{i-1}^{(t)} + a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{153}: a_i^{(t+1)} = 1 + a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{161}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$,

$W_{165}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{169}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} + a_{i-1}^{(t)}\, a_i^{(t)} + a_{i+1}^{(t)} \pmod 2$,

$W_{195}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} \pmod 2$,

$W_{225}: a_i^{(t+1)} = 1 + a_{i-1}^{(t)} + a_i^{(t)} + a_{i+1}^{(t)} + a_i^{(t)}\, a_{i+1}^{(t)} \pmod 2$.

5 Conclusiones

Se ha presentado una de las líneas de investigación que desarrolla el Grupo de Investigación en Autómatas Celulares y sus Aplicaciones, con relación al uso de los autómatas celulares lineales y su implementación como criptosistemas de cifrado en flujo. En particular se han estudiado las propiedades de los autómatas celulares de Wolfram como generadores de bits pseudoaleatorios, obteniéndose un clasificación de los mismos, que limita a 21 el número de posibles ACWs que poseen buenas propiedades pseudoaleatorias.

Referencias

[1] A. Fúster Sabater, D. de la Guía Martínez, L. Hernández Encinas, F. Montoya Vitini y J. Muñoz Masqué. *Técnicas criptográficas de protección de datos.* RA-MA, 2ª ed., Madrid, 2000.

[2] O. Goldreich and H. Krawczyk and M. Luby. *On the existence of pseudorandom generators.*

Lecture Notes in Computer Science vol.403. pp.57-75. 1990.

[3] S. W. Golomb. *Shift register sequences.* Holden-Day, San Francisco, 1967.

[4] D. de la Guía y A. Fúster. *Estudio de autómatas celulares para criptografía.* Actas de la IV Reunión Española de Criptología, pp. 167-174. 1996.

[5] D. de la Guía and A. Peinado. *Pseudorandom number generation based on nongroup cellular automata.* IEEE Proc. of 1999 International Carnahan Conference on Secury Technology, pp.370-376. 1999.

[6] D. de la Guía and A. Peinado. *On the sequences generated by* 90-150 *programmable cellular automata.* Proc. of 5th. SIC 2001, vol.VII. pp.492-495. 2001.

[7] L. Hernández Encinas y A. Martín del Rey y G. Rodríguez Sánchez. *Aplicaciones de los autómatas celulares a la generación de bits.* Bol. Soc. Esp. Mat. Apl. En prensa.

[8] P. D. Hortensius and R. D. McLeod and H. C. Card. *Parallel number generation for VSLI systems using cellular automata.* IEEE Trans. Comput. vol.38. pp.1466-1473. 1989.

[9] P. D. Hortensius and H. C. Card and R. D. McLeod and W. Pries. *Importance sampling for Ising computers using one-dimensional cellular automata.* IEEE Trans. Comput. vol.38. pp.769-774. 1989.

[10] J. C. Lagarias. *Pseudorandom number generators in cryptography and number theory.* Proc. Symp. in Appl. Math. vol.42. pp.115-143. 1990.

[11] W. Meier and O. Staffelbach. *Analysis of pseudo random sequences generated by cellular automata.* Advances in Cryptology-Proceedings of EUROCRYPT'91, Lecture Notes in Computer Science vol.547. pp.186-199. 1991.

[12] A. Menezes and P. van Oorschot and S. Vanstone. *Handbook of applied cryptography.* CRC Press, Boca Raton, FL., 1997.

[13] S. Micali and C. P. Schnorr. *Efficient perfect polynomial random number generators.* J. Cryptology vol.3. pp.157-172. 1991.

[14] S. Nandi and B. K. Kar and P. Chaudhuri. *Theory and applications of cellular automata in cryptography.* IEEE Trans. Comput. vol.43. pp.1346-1357. 1994.

[15] R. Schmitz. *Use of chaotic dynamical systems in cryptography.* J. Franklin Inst. vol.338. pp.429-441. 2001.

[16] M. Sipper and M. Tomassini. *Generating parallel random number generators by cellular programming.* Internat. J. Modern Phys. C vol.7. pp.181-190. 1996.

[17] M. Tomassini and M. Sipper and M. Zolla and M. Perrenoud. *Generating high-quality random numbers in parallel by cellular automata.* Future Generation Computer Systems vol.16. pp.291-305. 1999.

[18] A. J. Tomeu Hardasmal y R. Ruíz Rentero. *Cifrado de cadenas mediante autómatas celulares.* Actas de la IV Reunión Española de Criptología, pp.175-182. 1996.

[19] S. Wolfram. *Cellular automata.* Los Alamos Science vol.9. pp.2-21. 1983.

[20] S. Wolfram. *Cryptography with cellular automata.* Advances in Cryptology-Proceedings of CRYPTO'85, Lecture Notes in Computer Science vol.218. pp.429-432. 1986.