UNIÓN
REVISTA IBEROAMERICANA DE EDUCACIÓN MATEMÁTICA

ISSN: 1815-0640
Número 46. Junio 2016
Página 33-48

www.fisem.org/web/union
http://www.revistaunion.org

Autómatas celulares y aplicaciones

Alberto Cano Rojas, Ángela Rojas Matas

Fecha de recepción: 24/08/2015
Fecha de aceptación: 22/03/2016

Resumen	Un autómata celular es un modelo matemático para un sistema dinámico que evoluciona en pasos discretos. Este trabajo presenta una aplicación de los autómatas celulares para el cifrado de información y el reparto de secretos. Se detalla un ejemplo didáctico de su aplicación para el cifrado de secretos empleando imágenes digitales. El desarrollo de este trabajo ha servido como actividad académica dirigida a alumnado de Ingeniería en Informática, para fomentar su interés en la criptografía mediante herramientas matemáticas estudiadas a lo largo de su carrera.**Palabras clave:** autómatas celulares, criptografía
Abstract	Cellular automata are mathematical models for a dynamic system that evolve in discrete steps. This paper presents an application of cellular automata for encrypting information and sharing secrets. It is shown a didactic example of its application for encrypting secrets using digital images. This work has served as an academic activity aimed at students of Computer Engineering, to encourage their interest in cryptography by means of the use of mathematical tools learned along university courses.**Keywords**: cellular automata, cryptography
Resumo	Um autômato celular é um modelo matemático para um sistema dinâmico que evolui em passos discretos. Este trabalho apresenta uma aplicação de autômatos celulares para a encriptação de informação e partilha de segredos. É apresentado um exemplo didático da sua aplicação para encriptar segredos usando imagens digitais. O desenvolvimento deste trabalho serviu como atividade acadêmica destinada a estudantes de Engenharia Informática, para incentivar o seu interesse na criptografia mediante a utilização de ferramentas matemáticas avançadas que estudaram durante os seus cursos.**Palavras-chave:** autômatos celulares, criptografia

1. Introducción

Un autómata celular es un modelo matemático para un sistema dinámico que evoluciona en pasos discretos. Es adecuado para modelar sistemas naturales o artificiales que puedan ser descritos como una colección de objetos simples que interactúan localmente unos con otros.

Se aplican con éxito en:

- En el estudio de evolución de ecosistemas en biología
- En detección de fuegos en los bosques

- Simulación del tráfico en las ciudades
- Inteligencia Artificial

La primera etapa en la historia de los autómatas celulares se sitúa en los años 50 del siglo XX, con John von Neumann, matemático que tuvo una participación decisiva en la construcción del primer ordenador (el ENIAC en 1946). La segunda etapa llega a finales de los años 60 cuando Martin Gardner divulga en su columna científica *Mathematical Games* de la revista *Scientific American* el autómata celular conocido como "El Juego de la Vida" (Gardner, 1970, pp. 120-123). Se puede encontrar en Internet gran cantidad de información sobre ello. La tercera etapa llega a mediados de los años 80 con Stephen Wolfram, un científico reconocido además por ser el autor del popular programa de ordenador Mathematica (Wolfram, 2002).

Los autómatas celulares tienen, como se ha dicho anteriormente, múltiples aplicaciones. En este trabajo se verá cómo se pueden emplear para el cifrado de información y para el reparto de secretos.

La criptografía es un tema de indudable interés en la actualidad. Empresas y particulares demandan métodos cada vez más seguros para mantener a salvo de intrusos información confidencial. En este trabajo se verá cómo se pueden usar autómatas celulares para cifrar un mensaje de texto. En ocasiones puede que nos interese mantener en secreto una imagen digital. Se verá que también se pueden emplear autómatas celulares para ello.

Por otro lado, los esquemas de reparto de secretos son procedimientos criptográficos para compartir un secreto entre un conjunto de participantes de tal modo que solamente algunos de estos participantes puedan recuperar el secreto. Por ejemplo, supongamos que el director de un banco no quiere que la clave secreta que abre la caja blindada del banco sea conocida por ningún empleado. Por el contrario, desea autorizar a cinco empleados del banco, pero de modo que sólo cuando se junten al menos dos de ellos sí que se pueda abrir la caja fuerte. Esto es un ejemplo de un esquema de reparto de secretos (2, 5). En este trabajo se mostrará cómo podemos usar autómatas celulares para hacer reparto de secretos.

Los automátas celulares en su versión más sencilla, como la tratada en este artículo, y sin entrar en complicaciones criptográficas, es un tema de indudable interés como herramienta docente. Puede ser abordado por alumnos de la titulación de Ingeniería Informática de forma autónoma. Por otro lado, basta buscar por Internet bibliografía relacionada con el tema de automátas celulares para comprobar que se siguen publicando una gran cantidad de artículos en la actualidad, prueba de que los autómatas celulares siguen interesando a los investigadores (Bhasin, Kumar & Kathuria, 2013, pp. 355-357; Yampolskiy, Rebolledo-Mendez & Hindi, 2014, pp. 57-67).

Los métodos criptográficos desarrollados en este trabajo fueron implementados en Matlab por alumnos de primer curso de la titulación de Ingeniería Informática.

2. Autómatas celulares 1D

Se considera un vector. Cada componente del vector se va a llamar célula. Se supone que cada célula sólo puede tomar dos estados: cero (blanco) o uno (negro). Este tipo de autómata se conoce como autómatas celulares unidimensionales (1D) elementales.

Se va a efectuar un proceso dinámico donde se partirá de una configuración inicial $C^{(0)}$ de cada una de las células (etapa 0) y en cada nueva etapa se calculará el estado de cada célula de acuerdo al estado de las células vecinas y de la propia célula en la etapa anterior. Se indicará por $s_i^{(t)}$ el estado de la célula i-ésima en la etapa t.

Se considerará, por simplicidad, una vecindad de radio 1. Eso quiere decir que para calcular el estado de la célula i-ésima en la etapa t, es decir para calcular $s_i^{(t)}$, se usará el estado de las células i-1, i, i+1 en la etapa anterior, es decir, se usará $s_{i-1}^{(t-1)}$, $s_i^{(t-1)}$ y $s_{i+1}^{(t-1)}$. Un esquema se representa en la figura 1:

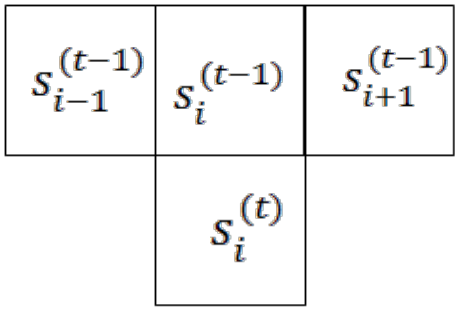


Figura 1: Esquema

La regla de evolución del autómata viene dada por el gráfico de la figura 2:

Figura 2. Regla de evolución del autómata

En este gráfico la fila superior indica el estado de la célula central (la célula i) y de sus dos células vecinas en una determinada etapa y, en la fila inferior, el estado de la célula central en la siguiente etapa. Por ejemplo, en el primer caso de la figura 2: si el estado de una célula en una determinada etapa es 1 (negro) y sus dos vecinas en esa misma etapa también están a 1 (negro) entonces el estado de la célula en la siguiente etapa será 0 (blanco). De la misma forma se interpretan el resto de los 7 casos.

Se considerará un autómata con *N* células que indicaremos por: s_0, s_1, … , s_{N-1}. Partiendo de una configuración inicial $C^{(0)}$ y con la regla de la figura 2 se podrá recalcular el estado en cada nueva etapa de las células s_1, … , s_{N-2}, pero se necesitan establecer unas condiciones en la frontera para poder recalcular s_0 y s_{N-1}. Es habitual considerar como condiciones en la frontera las establecidas en la figura 3 conocidas como *condiciones periódicas*.

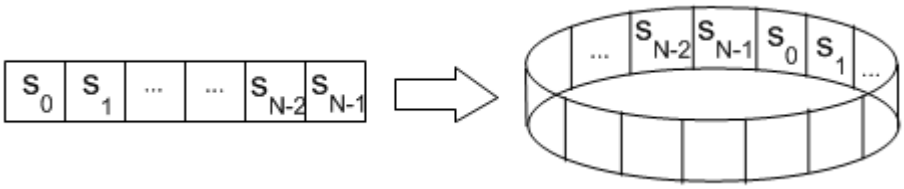


Figura 3: Condiciones periódicas en la frontera

Si el autómata se dibujara en una hoja de papel y se enrollara la hoja como se muestra en la figura 3, entonces, la vecindad a considerar para recalcular el estado de la primera célula s_0 en una nueva etapa será las células s_{N-1}, s_0, s_1 en la etapa anterior.

Análogamente el estado de la última célula s_{N-1} se recalculará teniendo en cuenta el estado de s_{N-2}, s_{N-1}, s_0 en la etapa anterior.

Por ejemplo, en la tabla 1 se considera un autómata con 4 células cuya configuración inicial fuera: $C^{(0)}$ ={1 1 1 0}. Aplicando la regla de la figura 2 con condiciones periódicas en la frontera se obtendrían los resultados de la tabla 1.

Etapa	$s_0^{(t)}$	$s_1^{(t)}$	$s_2^{(t)}$	$s_3^{(t)}$
0	1	1	1	0
1	1	0	0	0
2	1	1	0	1
…				

Tabla 1: Evolución de un autómata 1D con 4 células

Si indicamos por $Z_2 = \{0,1\}$, la regla de la figura 2 también se puede escribir como una aplicación $f : Z_2^3 \to Z_2$ tal que:

$$f(1,1,1) = 0$$
$$f(1,1,0) = 0$$
$$f(1,0,1) = 0$$
$$f(1,0,0) = 1$$
$$f(0,1,1) = 1$$
$$f(0,1,0) = 1$$
$$f(0,0,1) = 1$$
$$f(0,0,0) = 0$$

En una vecindad de radio 1, como es la considerada hasta ahora, hay 256 reglas posibles ya que hay 8 estados posibles de las tres células: {111, 110, 101, 100, 011, 010, 001, 000} y a cada uno de estos estados se le pueden asociar un cero o un uno, con lo que el número total de casos posibles es $2^8 = 256$.

En el caso de la regla considerada hasta ahora en el ejemplo estos valores son: 00011110 que en decimal se corresponde con: $00011110_2 = 30$. Por esta razón, la regla de evolución del autómata de la figura 2 se conoce como regla 30.

En la figura 4 se muestra la evolución de un autómata con la regla 30 y 43 células. En la configuración inicial todas las células están a cero salvo la célula 22 que está a 1. Se representan 21 etapas.

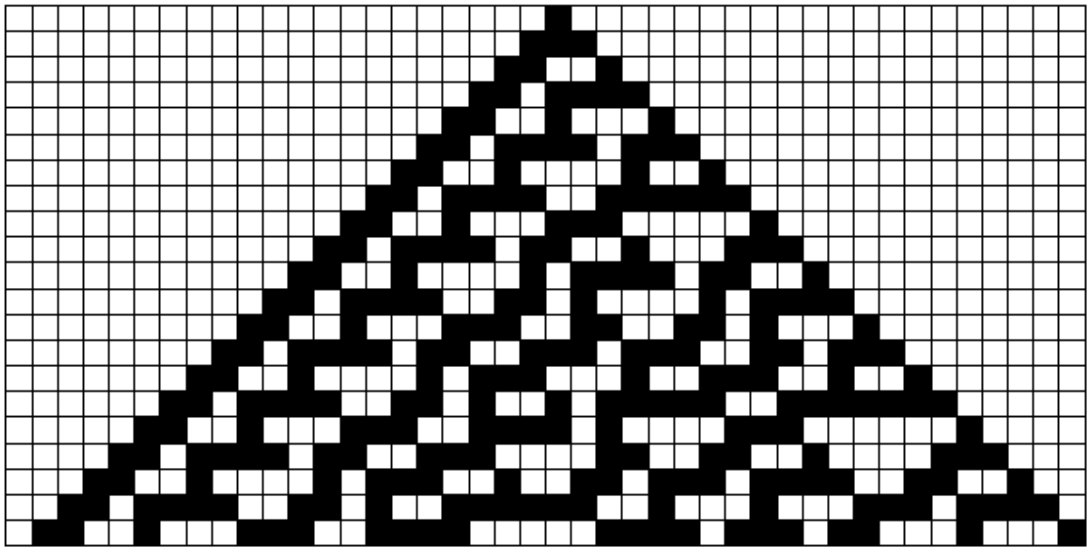


Figura 4: Evolución de la regla 30 con un autómata de 43 células

En la figura 5 se muestra la evolución de un autómata con la regla 30, donde el autómata tiene ahora 257 células y se muestran las 128 primeras etapas. La configuración inicial es: todas las células a cero salvo la central que está a uno que sería la primera fila (fila superior) de la figura 5.

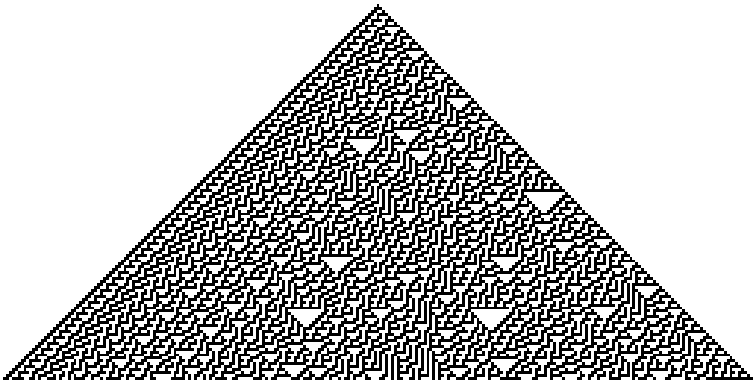


Figura 5: Regla 30 en un autómata de 257 células

En la figura 6 se muestra la similitud entre un detalle del caparazón de un caracol marino y un trozo de gráfico obtenido con la regla 30.

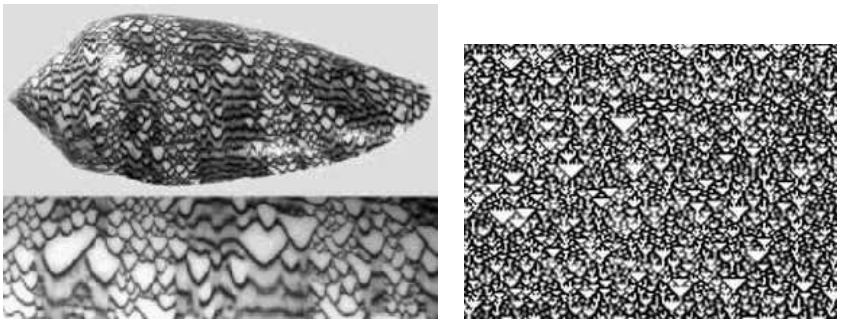


Figura 6: Caracol marino y regla 30

Los números pseudoaleatorios son muy importantes en Ciencias e Ingeniería. Stephen Wolfram (Wolfram, 1986a, pp. 123-169) fue el primero en proponer el autómata celular anterior como generador de números pseudoaleatorios. La idea sería partir de una configuración inicial del autómata, que sería la semilla del generador, efectuar unas cuantas iteraciones y quedarse con los bits generados por una determinada célula (por ejemplo, la central). De hecho, la orden "RandomInteger" del conocido programa Mathematica de Stephen Wolfram usa este autómata para la generación de números pseudoaleatorios.

En la figura 7, como curiosidad, se muestra la evolución del autómata celular correspondiente a la regla 90 en el que se parte de la misma configuración inicial que en el ejemplo anterior: todas las células a cero salvo la central que está a uno. Aparece el fractal conocido como triángulo de Sierpinski. Esta figura también se conoce como triángulo de Pascal módulo 2.

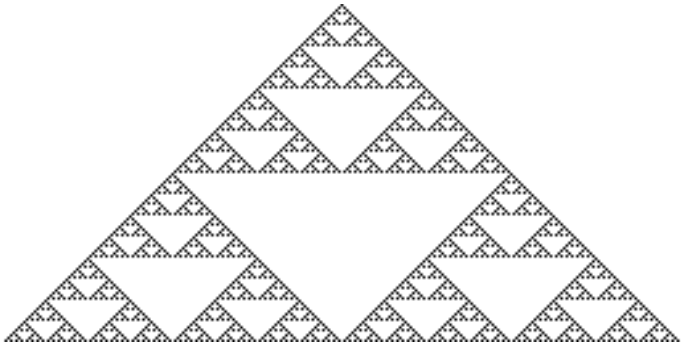


Figura 7: Autómata con la regla 90

Recordemos que el triángulo de Pascal se construía poniendo unos en los laterales del triángulo y un término del interior del triángulo se obtenía a partir de la suma de los dos que le precedían en la fila anterior, tal como se indica en la parte izquierda de la figura 8. A continuación tomamos módulo 2: los impares se ponen a 1 (negro) y los pares se ponen a 0 (blanco) tal y como se indica en la parte derecha de la figura 8.

Figura 8: Triángulo de Pascal y triángulo de Pascal módulo 2

Si repetimos esta idea para las 128 primeras filas del triángulo se obtiene la imagen de la figura 7. La relación entre la regla 90 y el triángulo de Pascal queda patente cuando se comprueba que la regla 90 es equivalente a la siguiente expresión:

$$s_i^{(t+1)} = s_{i-1}^{(t)} + s_{i+1}^{(t)} \pmod 2$$

El estado de la célula i en la etapa $t+1$ consiste en sumar (módulo 2) los estados de las células $i-1$ e $i+1$ en la etapa anterior, es decir, en la etapa t.

3. Cifrado con autómatas celulares 1D: cifrado de Wolfram

S. Wolfram (Wolfram, 1986b, pp. 429-432) también propuso usar autómatas celulares para cifrar un mensaje de texto con un método muy sencillo que se describe a continuación.

Se usará, por ejemplo, el alfabeto de la Tabla 2. Este alfabeto está compuesto de 32 caracteres que son: el símbolo #, las 27 letras del abecedario (desde la A hasta la Z) y los números del 1 al 4. Se podría elegir otro alfabeto por supuesto.

#	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	1	2	3	4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Tabla 2: Alfabeto de 32 caracteres

Por otro lado, un número entre 0 y 31 se escribe en binario con 5 bits: entre 00000 (correspondiente al 0) y 11111 (correspondiente al 31).

El mensaje que se quiere cifrar es, por ejemplo: “HOLA”.

Se convierte cada letra en número de acuerdo a la tabla anterior. Así, el mensaje da lugar a la secuencia: {8,16,12,1}.

Pasando a binario, con 5 bits cada uno, se obtendrá:

$$M=\{01000,10000,01100,00001\}$$

es decir, una secuencia de 20 bits.

Se escoge una clave secreta de cifrado que será la configuración inicial $C^{(0)}$ del autómata. Por ejemplo, la clave podría ser la palabra: “CAMINO”.

La clave se convierte en la secuencia {3,1,13,9,14,16} de acuerdo al alfabeto empleado. Se pasa a continuación cada número a binario con 5 bits: {00011,00001,01101,01001,01110,10000}. Entonces:

$$C^{(0)}=\{s_0^{(0)},s_1^{(0)},s_2^{(0)},\ldots,s_{N-1}^{(0)}\}=\{00011000010110101001011101000 0\}$$

Se aplica la regla 30 a un autómata que tiene $C^{(0)}$ como configuración inicial, efectuando un número determinado de iteraciones. Cuando se haya terminado el proceso, se elige una determinada célula y la secuencia de bits:

$$s_i^{(t)},s_i^{(t+1)},s_i^{(t+2)},\ldots$$

Por ejemplo, se elige i=15 y t=25. Como el mensaje a cifrar tiene 20 bits, se obtendría la siguiente secuencia de 20 bits:

$$K=\{s_{15}^{(25)},s_{15}^{(26)},\ldots,s_{15}^{(44)}\}=\{10010011101010000100\}$$

Resumiendo, lo que se ha hecho ha sido conseguir una lista K de 20 bits a partir de una clave secreta que era la palabra “CAMINO”. Se han obtenido los últimos 20 estados de la célula 15 (en este caso había 30 células y se numeraban de la forma 0, 1,…, 29).

A continuación, se va a ver cómo cifrar la secuencia de 20 bits correspondiente al mensaje original M con la clave K, que también tiene 20 bits. El método de cifrado

elegido es el conocido como método de Vernam. El cifrado Vernam es un algoritmo de criptografía inventado por Gilbert Vernam, ingeniero de los laboratorios AT&T, en 1917 y fue utilizado durante la segunda guerra mundial (Aguirre, 2006). Se trata de un método muy sencillo que usa la operación XOR.

La suma módulo 2 o suma en Z_2, reflejada en la tabla 3, también se llama suma XOR ya que coincide con la disyunción exclusiva usada en la lógica matemática (en inglés exclusive or), y es muy conocida por los informáticos.

+	**0**	**1**
0	0	1
1	1	0

Tabla 3: Suma módulo 2

Para cifrar el mensaje se hace una suma XOR bit a bit, que se indicará con el símbolo $\oplus$, o lo que es lo mismo, una *suma módulo 2 pero bit a bit*, entre las secuencias M y K, obteniendo:

D=M$\oplus$K=00110101011111111101

Cogiendo los bits de 5 en 5 de la secuencia D, pasando a decimal se obtiene {6,21,31,29} y, mirando el alfabeto, se convierte en: “FT42” que será definitivamente el mensaje cifrado.

Para descifrar se tendrán que deshacer los pasos anteriores.

Observar que si D=M$\oplus$K $\Rightarrow$ D$\oplus$K=M$\oplus$K$\oplus$K=M$\oplus\theta$=M

Ya que la suma XOR de una secuencia de bits K con ella misma es siempre una secuencia de ceros que se ha indicado por θ. Después, la suma XOR de una secuencia de bits M con una secuencia de ceros, da como resultado, evidentemente, la propia M. En resumen, M=D$\oplus$K, propiedad que se utilizará para descifrar.

El receptor del mensaje cifrado que reciba “FT42”, lo convertirá en números obteniendo {6, 21, 31, 29} que pasará a binario con 5 bits cada uno obteniendo la secuencia D. Como será conocedor de la clave K, hará M=D$\oplus$K, obteniendo de esta forma los 20 bits del mensaje original. A partir de ellos, se deducirá el mensaje de texto.

Este método de cifrado es muy sencillo. Es el clásico método de Vernam con la única particularidad de que la secuencia de bits pseudoaleatoria con la que se va a hacer la suma XOR se obtiene a través del uso de un autómata celular. Posteriormente otros autores propusieron variaciones del método anterior (Seredynski, Bouvry & Zomaya, 2004, pp.753-766).

3. Cifrado de una imagen usando un autómata celular 1D

En esta sección se verá cómo cifrar una imagen digital en lugar de un mensaje de texto. Se podría usar una variación del método descrito en la sección anterior, pero se llevará a cabo otro método distinto también basado en autómatas celulares.

En este trabajo se verá un algoritmo para cifrar una imagen que emplea autómatas celulares 1D. Es una versión simplificada del método propuesto por Martín, Rodríguez & de la Villa (2007, pp. 571-578).

Se considera un autómata elemental 1-D con N células: s_0, s_1, … , s_{N-1}. La vecindad va a ser de radio 1, por simplicidad (aunque los autores trabajan con un radio igual a 8).

Para actualizar en cada nueva etapa el estado de una célula se va a emplear una fórmula del tipo:

$$s_i^{(t)} = \sum_{p=-1}^{1} \lambda_p \, s_{i+p}^{(t-1)} + s_i^{(t-2)} = \lambda_{-1} \, s_{i-1}^{(t-1)} + \lambda_0 \, s_i^{(t-1)} + \lambda_1 \, s_{i+1}^{(t-1)} + s_i^{(t-2)} \pmod 2 \qquad (1)$$

Se trata de un autómata de orden 2, ya que el estado de una célula en la etapa t depende del estado del autómata en dos etapas anteriores: las etapas $t-1$ y $t-2$. El valor de cada parámetro λ_p puede ser cero o uno.

Ahora es necesario conocer la configuración del autómata en las dos primeras etapas: $C^{(0)}$ y $C^{(1)}$.

En la tabla 4 se muestra un ejemplo de la evolución de un autómata de 16 células donde se han elegido aleatoriamente las dos primeras configuraciones y donde se ha aplicado la fórmula (1) con: $\lambda_{-1}, \lambda_0, \lambda_1$ =101. Se han usado las condiciones periódicas de la figura 3.

Etapa	Configuración del autómata
0	0 1 1 1 0 1 1 0 1 1 0 0 0 1 1 0
1	1 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1
2	0 0 1 1 0 0 1 1 1 1 0 0 0 1 1 1
3	0 0 1 0 1 0 0 1 0 0 1 1 1 0 0 0

Tabla 4: Autómata 1D de orden 2

La fórmula (1) permite averiguar la configuración del autómata en la etapa t, es decir $C^{(t)}$, en función de las configuraciones en las dos etapas anteriores $C^{(t-1)}$ y $C^{(t-2)}$.

Como sumar y restar en Z_2 es lo mismo, eso quiere decir que la fórmula (1) se puede escribir también de la forma:

$$s_i^{(t-2)} = \sum_{p=-1}^{1} \lambda_p \, s_{i+p}^{(t-1)} + s_i^{(t)} \pmod 2 \quad (2)$$

Esta fórmula indica que podemos dar marcha atrás y recuperar el valor de la configuración $C^{(t-2)}$, si se conocen las configuraciones $C^{(t-1)}$ y $C^{(t)}$. Por esta razón, este tipo de autómatas se llaman *autómatas reversibles*.

Esta es la idea que permitirá desarrollar un método de cifrado de una imagen digital. Se aplicará esta técnica a una imagen en escala de grises.

La imagen secreta que se va a cifrar se muestra en la figura 9. Es una imagen de tamaño 256×256 en escala de grises, aunque el método se puede adaptar fácilmente al caso de imágenes en color. Esta imagen tiene 256 niveles de gris posibles para cada píxel que van desde 0 correspondiente al negro hasta 255 correspondiente al blanco.

Una imagen en escala de grises no es más que una matriz de números donde cada número indica el nivel de gris del píxel: 0 (en binario con 8 bits es la secuencia 00000000) que se corresponde con el negro y 255 (en binario con 8 bits es la secuencia 11111111) que se corresponde con el blanco. Son necesarios 8 bits (1 byte) para almacenar el nivel de gris de un píxel de la imagen de la figura 9. Este rango de niveles de gris es muy habitual en las imágenes que circulan por Internet.

Figura 9: Imagen secreta

Cada fila tendrá 256 píxeles, lo que hace un total de 256×8=2048 bits. Se cogerán las filas de dos en dos. Pues bien, los 2048 bits de la primera fila serán el estado inicial $C^{(0)}$ de un autómata y análogamente los 2048 bits de la segunda fila serán el estado $C^{(1)}$. Después se efectúan una serie determinada de iteraciones T, obteniendo las sucesivas configuraciones del autómata $C^{(2)}$, $C^{(3)}$, …, $C^{(T-1)}$, $C^{(T)}$.

Pues bien $C^{(T-1)}$ y $C^{(T)}$ serán dos vectores con 2048 bits cada uno. Se pasa a decimal cada grupo de 8 bits y se obtendrán dos vectores de 256 componentes que

serán las dos primeras filas de la imagen cifrada. De esta misma forma se van cogiendo de dos en dos las sucesivas filas de la imagen original.

La clave secreta será los números $\lambda_{-1}, \lambda_0, \lambda_1$ y el número de iteraciones T. El resultado, con $\lambda_{-1}, \lambda_0, \lambda_1 = 101$ y $T = 5$, se muestra en la figura 10.

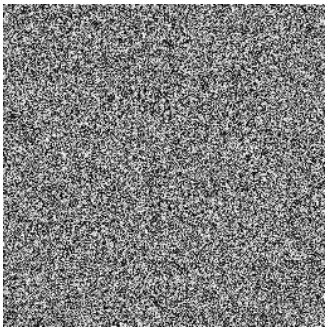


Figura 10: Imagen cifrada

La versión simplificada que se ha visto anteriormente tiene interés sólo desde el punto de vista docente. Para hacer el método más seguro se deberían usar claves distintas para cada par de filas de la imagen. También es aconsejable efectuar previamente una permutación aleatoria de los píxeles de la imagen.

Como el autómata es reversible, basta con dar marcha atrás para recuperar la imagen original para lo que hará falta saber, lógicamente, la clave secreta.

El mismo grupo de alumnos que implementó la técnica desarrollada en esta sección propuso una pequeña variación de la misma como técnica de cifrado. Teniendo en cuenta que en la asignatura de Matemática Discreta de la titulación habían trabajado el tema de aritmética modular, pensaron, y de hecho es así, que se podía simplificar aún más el método si:

- cada celda podía tomar valores entre 0 y 255
- se trabaja, lógicamente, con aritmética módulo 256 para que los niveles de gris permanezcan siempre confinados dentro del rango 0-255.

Efectivamente en este caso el paso de binario a decimal y al revés no es necesario, lo que simplifica considerablemente el código. Ahora bien, la fórmula (1) quedaría ahora de la siguiente forma:

$$s_i^{(t)} = \sum_{p=-1}^{1} \lambda_p \, s_{i+p}^{(t-1)} + s_i^{(t-2)} \pmod{256} \quad (3)$$

A la izquierda de la figura 11 se muestra la imagen original: se trata de una imagen de tamaño 136x136 con niveles de gris en el mismo rango de antes, es

decir, entre 0 y 255. A la derecha de la figura 11 Se muestra la imagen cifrada obtenida para $\lambda_{-1}, \lambda_0, \lambda_1 = 25,16,134$ y $T = 5$.

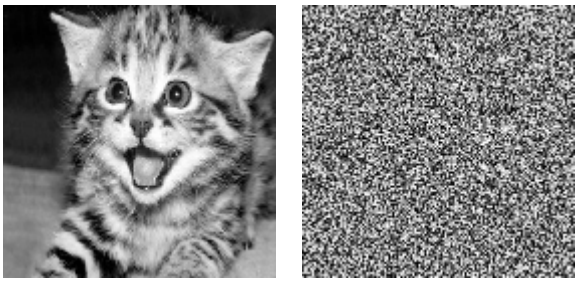
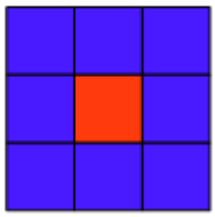


Figura 11: Imagen original a la izquierda e imagen cifrada a la derecha

5. Autómata celulares 2D y el reparto de secretos

Una idea similar se puede aplicar en dos dimensiones (Álvarez, Hernández & Martín, 2005, pp. 411-418).

En este caso el autómata es una matriz M×N donde cada elemento de la matriz es una célula con dos posibles estados: 0 y 1. En un autómata 2D una célula se indicará ahora de la forma $s_{(i,j)}^{(t)}$ donde i es la fila, j es la columna y t es la etapa. De nuevo, el estado de una célula en una etapa dependerá del estado de las células vecinas en etapas anteriores. Se pueden considerar distintos tipos de vecindades, por ejemplo, la que se indica en la figura 12:



$s_{(i-1,j-1)}^{(t)}$	$s_{(i-1,j)}^{(t)}$	$s_{(i-1,j+1)}^{(t)}$
$s_{(i,j-1)}^{(t)}$	$s_{(i,j)}^{(t)}$	$s_{(i,j+1)}^{(t)}$
$s_{(i+1,j-1)}^{(t)}$	$s_{(i+1,j)}^{(t)}$	$s_{(i+1,j+1)}^{(t)}$

Figura 12: Una célula en la posición (i,j) (en rojo) y sus 8-vecinas (en azul)

Los autómatas 2D también se pueden usar como generadores de números pseudoaleatorios y como herramientas de cifrado de una forma sencilla y similar a la explicada en los apartados anteriores. Sin embargo, se mostrará a continuación una sencilla aplicación a un protocolo criptográfico conocido como el *reparto de secretos* ya que se desea introducir a los alumnos en esta técnica criptográfica por su indudable interés.

El secreto será una imagen digital en escala de grises como la imagen de la figura 9. Se verá cómo funciona el esquema (2, 2) para esta imagen. En este caso, hay dos participantes en el proceso y solamente cuando se junten los dos podrán recuperar el secreto (la imagen digital de la figura 9). El proceso a seguir es muy sencillo y se explica a continuación.

Se pasa cada nivel de gris de un píxel a binario. De manera que la primera fila que tiene 256 píxeles se convertirá en una secuencia de 256x8=2048 bits. Como la imagen empleada tiene 256 filas, se obtendrá entonces una matriz de tamaño 256×2048 de ceros y unos. La matriz binaria anterior será la configuración inicial $C^{(0)}$ de un autómata 2D. Después se creará una matriz pseudoaleatoria de ceros y unos exactamente del mismo tamaño, es decir 256×2048, que será la configuración $C^{(1)}$. A continuación, se pone el autómata a trabajar usando la siguiente fórmula:

$$s_{(i,j)}^{(t)} = \sum_{k,p=-1}^{1} \lambda_{k,p}\, s_{(i+k,j+p)}^{(t-1)} + s_{(i,j)}^{(t-2)} \pmod 2 \qquad (4)$$

Hay 9 coeficientes $\lambda_{k,p}$ en la fórmula anterior. Cada uno puede ser un cero o un uno, por lo tanto, hay $2^9 = 512$ posibilidades. Se usarán condiciones periódicas tanto para las filas como para las columnas.

Entonces se efectúan una serie de iteraciones, por ejemplo, siete. Se seleccionan $C^{(6)}$ y $C^{(7)}$. Ambas son matrices binarias de tamaño 256×2048 que pasaremos a decimal agrupando los bits de 8 en 8. De esta forma se obtendrán dos matrices S_1 y S_2 de tamaño 256×256 cuyos elementos varían entre 0 y 255 que se podrán mostrar como imágenes en escala de grises. En la figura 13 se muestran las imágenes S_1 y S_2 obtenidas usando los valores de $\lambda_{k,p}$ de la Tabla 5.

$\lambda_{-1,-1}$	$\lambda_{-1,0}$	$\lambda_{-1,1}$
$\lambda_{0,-1}$	$\lambda_{0,0}$	$\lambda_{0,1}$
$\lambda_{1,-1}$	$\lambda_{1,0}$	$\lambda_{1,1}$

=

1	1	1
1	0	1
1	0	0

Tabla 5: Valores de $\lambda_{k,p}$ usados

Pues bien, al primer participante del esquema (2, 2) se le dará S_1 y al segundo participante se le dará S_2. Ninguno de ellos tendrá ninguna información sobre la imagen secreta ya que la información que cada uno posee es una imagen con aspecto aleatorio como se muestra en la figura 13. Sin embargo, cuando ambos se junten, como el autómata dado en la ecuación (4) es también reversible, podrán retroceder y calcular $C^{(0)}$, es decir, la imagen secreta original.

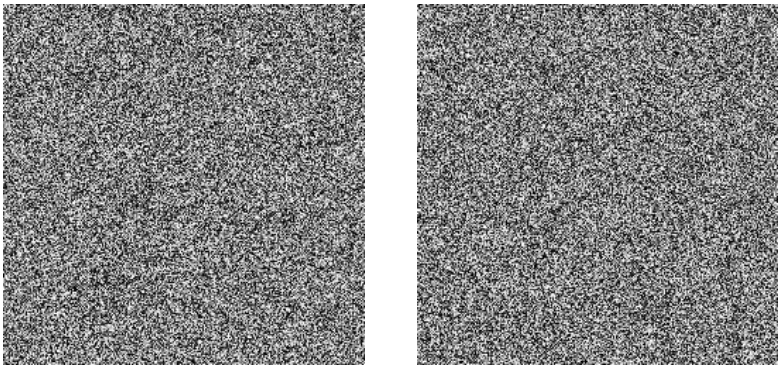


Figura 13: Imágenes de cada participante en el reparto de secretos (2,2)

Esta idea se puede generalizar para otros esquemas de reparto de secretos. Por ejemplo, si se desea hacer un esquema (2, 5): de nuevo la imagen original se haría coincidir con el estado inicial del autómata $C^{(0)}$, la configuración $C^{(1)}$ se escogería aleatoriamente y se harían un número determinado de iteraciones, por ejemplo, siete. Entonces las imágenes S_1, S_2, S_3, S_4 y S_5 se obtendrían a partir de las 5 últimas configuraciones $C^{(3)}, C^{(4)}, C^{(5)}, C^{(6)}$ y $C^{(7)}$, respectivamente. Si se juntan dos participantes con configuraciones consecutivas podrán dar marcha atrás y recuperar la imagen secreta.

En este método también se puede trabajar con congruencias módulo 256, lo que supone una simplificación muy notable del código.

6. Conclusiones

Los autómatas celulares son un tema de indudable interés para un futuro ingeniero informático debido a las interesantes aplicaciones que tienen. Se ha visto en este trabajo cómo se pueden aplicar para cifrar texto, para cifrar una imagen y también para un esquema de reparto de secretos donde el secreto es una imagen digital.

Las ideas han surgido de dos artículos de investigación (Álvarez et al., 2005, pp. 411-418 y Martín et al., 2007, pp. 571-578). Se propusieron a alumnos de la titulación de Ingeniería Informática una implementación de dichas ideas, aunque en versiones simplificadas. Se ha acercado así la investigación a la docencia universitaria.

Por otro lado, los alumnos han usado Matlab para implementar las técnicas aquí descritas. Esto les sirve de paso para conocer y afianzar el lenguaje de programación de Matlab que se está convirtiendo en una herramienta de gran utilidad para un ingeniero.

Los resultados fueron interesantes porque consiguieron una gran motivación en los alumnos que opcionalmente eligieron llevar a cabo esta experiencia.

Bibliografía

Álvarez, G., Hernández, L., Martín, A. (2005). *A new secret sharing scheme for images based on additive 2-Dimensional cellular automata*. Lecture Notes in Computer Science, 3522, 411-418.

Bhasin, H., Kumar, R., Kathuria, N. (2013). *Cryptography Using Cellular Automata*. International Journal of Computer Science and Information Technologies, 4 (2),355-357.

Aguirre, J.R. (2006). *Material de Seguridad Informática y Criptografía*. Recuperado el 22 de marzo de 2016, de http://www.deic.uab.es/material/26118-09CifraClasica.pdf

Gardner, M. (1970). *Mathematical Games – The fantastic combinations of John Conway's new solitaire game "life"*. Scientific American, 223, 120–123.

Martín, A., Rodríguez, G., de la Villa, A. (2007). *A protocol to cipher digital images based on cat maps and cellular automata*. Lecture Notes in Computer Science, 4477, 571-578.

Seredynski, F., Bouvry, P., Zomaya, P. (2004). *Cellular automata computations and secret key cryptography.* Parallel Computing, 30, 753-766.

Wolfram, S. (1986,a). *Random sequence generation by cellular automata*. Advances in Applied Mathematics, 7, 123-169.

Wolfram, S. (1986,b). Cryptography with cellular automata. Lecture Notes in Computer Science, 218, pp. 429-432.

Wolfram, S. *A New Kind of Science*. Wolfram Media, Inc., 2002.

Yampolskiy, R.V, Rebolledo-Mendez, J. D., Hindi, M.M. (2014). *Password Protected Visual Cryptography via Cellular Automaton Rule 30*. Transactions on Data Hiding and Multimedia Security IX, Lecture Notes in Computer Science, 8363, 57-67.

Autores:

Alberto Cano Rojas. Doctor en Informática por la Universidad de Granada, España. Actualmente es Assistant Professor en la Virginia Commonwealth University, EEUU. Su labor se centra en la investigación de Sistemas Inteligentes y Modelos Computacionales. Interesado en las Matemáticas recreativas y sus aplicaciones a la docencia universitaria en enseñanzas técnicas en ingeniería Informática. acano@vcu.edu

Ángela Rojas Matas. Licenciada en Matemáticas, Doctora en Informática. Profesora del Departamento de Matemáticas de la Universidad de Córdoba, España. Durante muchos años ha impartido la asignatura Álgebra Lineal en el primer curso de la titulación universitaria de Ingeniería Informática. angela.rojas@uco.es