UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

ALGORITMOS GENÉTICOS PARA LA OPTIMIZACIÓN DE CIFRADO DE IMÁGENES MEDIANTE SISTEMAS CAÓTICOS

TESIS

QUE PARA OBTENER EL TÍTULO DE:

LIC. MATEMÁTICAS APLICADAS Y COMPUTACIÓN

PRESENTA:

JESSICA LECHUGA RAMOS

ASESOR:

DRA. KATYA RODRÍGUEZ VÁZQUEZ

Santa Cruz Acatlán, Estado de México, 2021

Dedicado a mi familia: Mis hermanas Fany y Diana, a mi mamá Gloria, a mi abuelita Meche y finalmente a mi esposo Ángel.

El alma es el espejo de un universo indestructible.
Gottfried Wilhelm Leibniz

Agradecimientos

Jamás pensé que la fuerza de una familia fuese tan grande, que te permite superar cualquier adversidad; por ello, le agradezco a todos los miembros de mi familia que me han apoyado, me han regalado su paciencia y su compresión en todo momento, sobre todo a:

Mis tíos por demostrarme que están en cualquier instante que los necesito y ser mi fuente de inspiración.

Mi abuelita que siempre estuvo en cada etapa de mi vida, apoyándome, demostrándome su gran fortaleza, a quien admiro, respeto y sobre todo guardo un gran cariño.

Diana mi hermana menor, por su ternura, paciencia, solidaridad y sobre todo su lealtad, lo que poco se encuentra en una persona. Me ha ayudado en los momentos menos esperados, que, aunque no lo admita en voz alta sé que me admira y sé que es una de las personas que siempre me extenderá la mano en cualquier momento.

Fany mi hermana mayor, quien siempre ha sido mi gran compañera de vida y quien me brinda sin dudar su gran amistad, lealtad, comprensión y sobre todo guía, quien conozco de toda la vida y quien desde siempre he admirado y he tratado de seguir sus pasos, en todo momento me ha brindado todo su apoyo no siendo este proyecto una excepción pues ha sido una pieza clave para su realización.

Mi madre, quien me educo, cuido y me dio mucho más de lo que una persona le puede ofrecer a su creación (como suele decir), porque a pesar de lo que suele pensar es una excelente madre, puesto que me ha dado las lecciones más importantes de mi vida con tanta paciencia y amor, porque fue quien siempre creyó en mí y me demostró que se puede superar cualquier cosa.

Mi gran amigo Gerardo que me ayudo con el entendimiento de algunos temas pertinentes para el proyecto, él que siempre me escucho cuando estaba en un lío y siempre me brindo ese gran apoyo.

La doctora Katya que a través del tiempo siempre mantuvo sus puertas abiertas y su gran apoyo como asesora y persona en todo este proyecto.

Mi ex profesor Samuel Ornelas que siempre escucho todos mis problemas y con la paciencia que lo caracteriza, brindarme grandes consejos y siempre con la disposición de ayudar.

Finalmente, a mi esposo quien estuvo en todo momento en este proceso levantándome cuando lo necesitaba y apoyando cada decisión que tome para finalizar esta tesis así como su gran comprensión y la fuerza que me brindo sin dudarlo una sola vez, convirtiéndose en ese refugio que necesito y la determinación que me alimenta.

Todas y cada una de estas personas han jugado un papel importante para la realización de este trabajo y en verdad les estoy muy agradecida por todo lo que me han brindado y enseñado.

Lo que llamamos "casualidad" no es más que la ignorancia de las causas físicas.
Gottfried Wilhelm Leibniz

Índice general

Índice de figuras

Índice de tablas

Índice de abreviaturas

ADN Ácido Desoxirribonucleico.

AG Algoritmo Genético.

AP Algoritmo Propuesto.

ARN Ácido Ribonucleico.

ASCII American Standard Code for Information Interchange.

CCP Coeficiente de Correlación de Pearson.

F False.

GB Gygabyte.

GHz Gigaherz.

Gnr Generación.

Hrz Horizontal.

IA Inteligencia artificial.

Pc Parámetro de cruza.

Pch Parámetro de cambio.

Pm Parámetro de mutación.

PRNG Pseudorandom Number Generator.

RAE Real Academia Española.

RAM Random Access Memory.

T True.

XOR Exclusive OR.

Capítulo 1

Introducción

Desde hace más de 4 mil años el hombre ha hecho uso de la escritura secreta que se solía implementar como un sistema seguro de comunicación. Este método surge de la necesidad de enviar mensajes sin que se supiera de su existencia y de esta manera no saber que se quiere transmitir. Precisamente esto los llevó a esconder el mensaje de diferentes maneras; a este proceso se le conoce como esteganografía. Un ejemplo de ello era rapar al mensajero, escribir el contenido en su cuero cabelludo y posteriormente dejar crecer su cabello (Singh, 2000, pág.16). Las personas que hacían esto podían permitir que el tiempo trascurriera pues la urgencia de transmitir el mensaje no era inmediata. De esa manera, al llegar al receptor, simplemente se rapaba y entregaba el secreto; de esta forma el mensajero podía viajar hasta el destino de forma segura. Sin embargo, tiene un punto débil, si se descubre el mensaje en el trayecto queda expuesto en el momento. Es por ello, que paralelamente a este se desarrolló la criptografía, el punto a comparar de la esteganografía es que el cifrado no trata de ocultar la existencia de un mensaje, sino su contenido. Con este método, aunque un tercero obtenga el mensaje no sabrá interpretarlo. Para ocultar la información del mensaje se requiere de un proceso llamado codificación.

Un método sencillo de criptografía que se remonta en la antigua Roma fue utilizado por Julio César, el cual consiste en implementar el desplazamiento de las letras del alfabeto original, se sustituye por otra letra que se localiza a un número n de posiciones más adelante en el alfabeto (Singh, 2000, pág.20). Por ejemplo, si se le da a n la asignación del número 3 se puede ver el método en la siguiente figura:

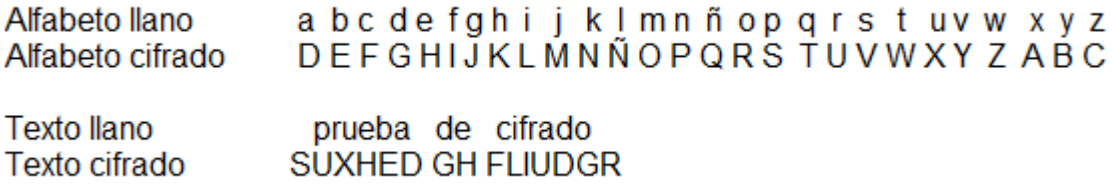


Figura 1.1: Método de César
(Elaboracion propia)

Este arte evolucionó por la necesidad de métodos más seguros puesto que surgió el criptoanálisis; el cual se dedica a estudiar los Sistemas Criptográficos con el fin de encontrar vulnerabilidades. Con el paso del tiempo la tecnología influyo en los métodos de criptografía de forma que son utilizados a diario en la transferencia de datos.

A lo largo de la historia la experiencia ha demostrado que los mecanismos de criptografía deben ser públicos y no secretos, esto se debe a que si un atacante encuentra alguna vulnerabilidad en el mecanismo es probable que no dé a conocer el fallo de este o incluso no se sepa que tiene alguna forma de romper tal mecanismo. Todo esto implica que lo único que se debe ocultar es la clave, es por lo que al pensar en las propiedades de los Sistemas Caóticos, (que más adelante veremos en el capítulo 3) lo hace ideal para esta disciplina si se toma en cuenta que el hardware mejora y adquiere más poder computacional, aumenta la posibilidad de romper la seguridad de un sistema mediante un simple ataque de fuerza bruta siendo más factible. Además, el criptoanálisis evoluciona a la par de la criptografía y ahora poniendo en juego el poder computacional actual abriendo paso a múltiples posibilidades que en nuestros días, muchas cosas se hacen posibles que en tiempos donde se inició la criptografía ni siquiera se pensaba.

Al pensar en la vida cotidiana nos podemos dar cuenta que la multimedia pasa a ser parte de nuestros medios de comunicación que se emplea de forma personal y publica. Es un término que se emplea en los sistemas u objetos que se utilizan en diferentes medios para presentar o trasmitir información combinando de manera simultánea: textos, imágenes, audios, entre otros. Por su parte una imagen se comprende matemáticamente como una función $f(x, y)$ bidimensional, donde x y y corresponden a un par de coordenadas espaciales de f. Por otro lado el valor de f representa la intensidad o nivel de gris de la imagen en la coordenada (x, y) correspondientes; cada uno de estos elementos son llamados pixeles (Gonzales and Woods, 2002, pág.1).

Las imágenes que son parte de la multimedia son un medio importante actualmente, de forma que en muchas ocasiones son una necesidad en cualquier medio digital. En la actualidad la encriptación de imagen y video tiene aplicaciones en diversos campos como: comunicaciones inalámbricas, sistemas multimedia, imágenes médicas, telemedicina y comunicaciones militares. Existen imágenes con información personal que deseamos enviar y que ningún otro a excepción del receptor queremos que sean mostradas. Por ejemplo la información que resguardan los hospitales; en esta se contiene información sensible del paciente, es por ello que se debe dar garantía de la integridad de la misma. Por este motivo es importante tener métodos que protejan la información que contienen este tipo de archivos. Sin embargo, los métodos convencionales que se considera que tienen cifrado fuerte que se usan para el texto no son factibles para las imágenes ya que estas tienen diferentes propiedades que los textos no tienen. Sabiendo esto, lo que se ha hecho es modificar estos algoritmos para que trabajen con las imágenes o se buscan otros enfoques adecuados para esta tarea. En esta tesis, propongo analizar un nuevo enfoque con el Algoritmo Genético (AG) que pertenece a una de las líneas de la Inteligencia artificial (IA) que pueden trabajar muy bien junto con los Sistemas Caóticos para mejorar el cifrado de imagen utilizando cualquier sistema para ello.

1.1. Objetivos

1.1.1. General

Crear un Algoritmo Genético cuyo propósito será optimizar un cifrado de imagen, que ha sido generado mediante un sistema de cifrado utilizando una ecuación de carácter caótico.

1.1.2. Específicos

- Analizar el método de cifrado utilizando una función caótica.
- Analizar la optimización que el AG proporciona mediante métricas de evaluación (criptoanálisis).
- Estudiar el comportamiento del AG ante el modelo cifrado.
- Cotejar el cifrado por la función caótica contra la optimización del Algoritmo Propuesto (AP).
- Proporcionar una forma para descifrar la imagen final.
- Demostrar que un AG puede optimizar un cifrado de imagen independientemente del método de criptografía que se utilice.
- Ajustar los parámetros del AG para tener un buen resultado ante cualquier prueba.

1.2. Hipótesis

- Independientemente del método o sistema que se utilice para cifrar, se puede optimizar el cifrado de imagen mediante Algoritmos Genéticos (AGs).

1.3. Metodología

El proceso llevado a cabo para elaborar el trabajo está compuesto de varios pasos a seguir que llevan al desenlace del presente trabajo. Este empieza por la investigación de diversos temas que se recuperaron de datos sobre distintos tipos de cifrado de imagen llevados a cabo en investigaciones que fueron escritos en algunos artículos académicos, páginas web y libros. Para profundizar en los temas, se procedió a la lectura sobre los temas que corresponden a Sistemas Caóticos y sus propiedades, líneas de IA que pueden ser una herramienta valiosa a diversas aplicaciones, así como a familiarizarse un poco con el proceso que se lleva a cabo con las imágenes. Cabe mencionar que en los medios de investigación se encontró algunos métodos de cifrado que utilizaban Algoritmos Genéticos (AGs)y sistemas caóticos, incluso ambas como propuestas nuevas así como diversas comparaciones de algunos métodos muy interesantes que se tomo en cuenta para el trabajo.

Una vez adentrados en los temas que se necesitan, el establecimiento de objetivos fue el paso a seguir junto con el planteamiento del problema e hipótesis. En esta parte se necesitó tener en claro las limitaciones que existen en el proyecto así como el alcance que lleva el trabajo, esto es para tener cuidado en no extender el trabajo a temas que no competen con el objetivo propuesto e ir en direcciones distintas a las que se plantearon, por lo que esta parte fue crucial en la tesis.

Teniendo las bases anteriores se propuso cada parte del AG, sobre todo el pináculo de este, la función objetivo. Planteados los elementos del algoritmo, se tomó en cuenta que solo era parte del sistema propuesto, quiere decir, que se debían contemplar algunas consideraciones, las cuales consisten en estimar el método de descifrado y el método por el cual la información arrojada del AG se enviaría al receptor.

Antes de proceder a la elaboración del programa, se estableció en que lenguaje y versión así como las herramientas que se utilizarían para programar. Una vez contemplado, se procedió a crear el algoritmo; a lo largo del proyecto se hicieron 14 versiones del mismo, en razón de que surgieron otras características que no se contemplaron. Se estuvo adaptando para poder lograr el objetivo planteado. De esta manera se logró crear la versión final.

Con el algoritmo en funcionamiento solo restaba hacer varias pruebas exhaustivas tomando para ello un conjunto de imágenes que ayudaron a ajustar los parámetros del algoritmo. De ese modo se pudo medir su efectividad, cotejar el resultado contra el cifrado original y evaluar en diferentes aspectos los resultados arrojados. Con los resultados se pudo concluir si el algoritmo había alcanzado los objetivos específicos y el general, ver si la hipótesis planteada se cumplía, delimitar a que cierto tipo de problema se puede aplicar para mayor efectividad y tomar algunas consideraciones futuras para el trabajo.

1.4. Estructura de la tesis

- **Capítulo 2 Algoritmo Genético**: Se desglosa conceptos necesarios para el entendimiento de los Algoritmos Genéticos; los conceptos descritos para el entendimiento de los Algoritmo Genéticos (AGs) son la base del desarrollo del proyecto. Con esto se busca que se comprenda la función que llevarán a cabo así como su estructura.
- **Capítulo 3 Sistemas Caóticos**: Se definen los conceptos esenciales de los sistemas caóticos en el desarrollo del trabajo, estos se implementarán en la parte del cifrado de la imagen donde se optimizará el resultado.
- **Capítulo 4 Criptografía**: Se repasan las definiciones de la criptografía y los métodos que se pueden usar para el cifrado de las imágenes, así como, el planteamiento del medio por el cual se cifraran estas mismas.
- **Capítulo 5 Algoritmo Propuesto para la Optimización del Cifrado de Imágenes**: Se integran los temas anteriores en donde se desglosa un algoritmo que permita contemplar todos sus elementos para optimizar el cifrado de una imagen.
- **Capítulo 6 Resultados**: Se presenta los resultados que fueron arrojados durante el desarrollo del trabajo así como las observaciones que se obtuvieron, con ello se explica los puntos relevantes que se dieron durante las ejecuciones del algoritmo.
- **Capítulo 7 Conclusiones**: Se exponen las conclusiones derivadas de la investigación que se llevo a cabo del algoritmo así como las aportaciones que surgieron a lo largo del proyecto y recomendaciones para investigaciones posteriores.

Capítulo 2

Algoritmos Genéticos

2.1. Introducción

A lo largo del tiempo se han desarrollado diferentes técnicas de búsqueda como se muestra en la siguiente imagen (figura 2.1), la cual visualiza un esquema donde se puede observar que los Algoritmos Genéticos son parte de estas técnicas de búsqueda aleatoria guiada y a su vez provienen de los algoritmos evolutivos que desprenden otros tres paradigmas: Programación Evolutiva, Estrategia Evolutiva y Programación Genética. Los AGs son los más extensos de entre estos paradigmas mostrando ser los más populares en el campo.

Los algoritmos fueron creados en 1975 por John Henry Holland junto con sus estudiantes y colegas en la Universidad de Michigan. Las principales razones por las que los crearon fueron para explicar los procesos de adaptación de los sistemas naturales y diseñar un sistema artificial que contenga los mecanismos de los mismos sistemas naturales. Lo que no se imaginaron fue el éxito que tendría y la amplia variedad de aplicaciones que han sido implementados para diferentes campos que han sido llevados a grandes descubrimientos.

La idea básica de la evolución biológica, la selección natural propuesta por Charles Darwin, es la clave para poder entender los AGs puesto que John Holland la tomó como base principal para los mismos. De acuerdo con el principio de la selección natural, "supervivencia del más apto", los individuos compiten por recursos en su ámbito, incluso entre su especie para atraer a un compañero y así reproducirse, los seres vivos que se adapten mejor a su entorno tienen más probabilidades tanto de sobrevivir como de reproducirse, con ello tendrían la posibilidad de generar mayor descendencia y mejor propagación de sus genes mediante el cual puede transmitir las mismas cualidades que posee o incluso generar mejores. Esto podría implicar un mejor resultado tendiendo a una mejor aptitud que la del progenitor. Por el contrario, aquellas que no puedan adaptarse ante su entorno se convertirán en escasas y la reducción será tan notable que tenderá a la extinción. De este modo la especie con menor adaptabilidad tendrá menos oportunidad a reproducirse y como consecuencia tendrá poca descendencia, por lo que hay una lucha continua por la vida.

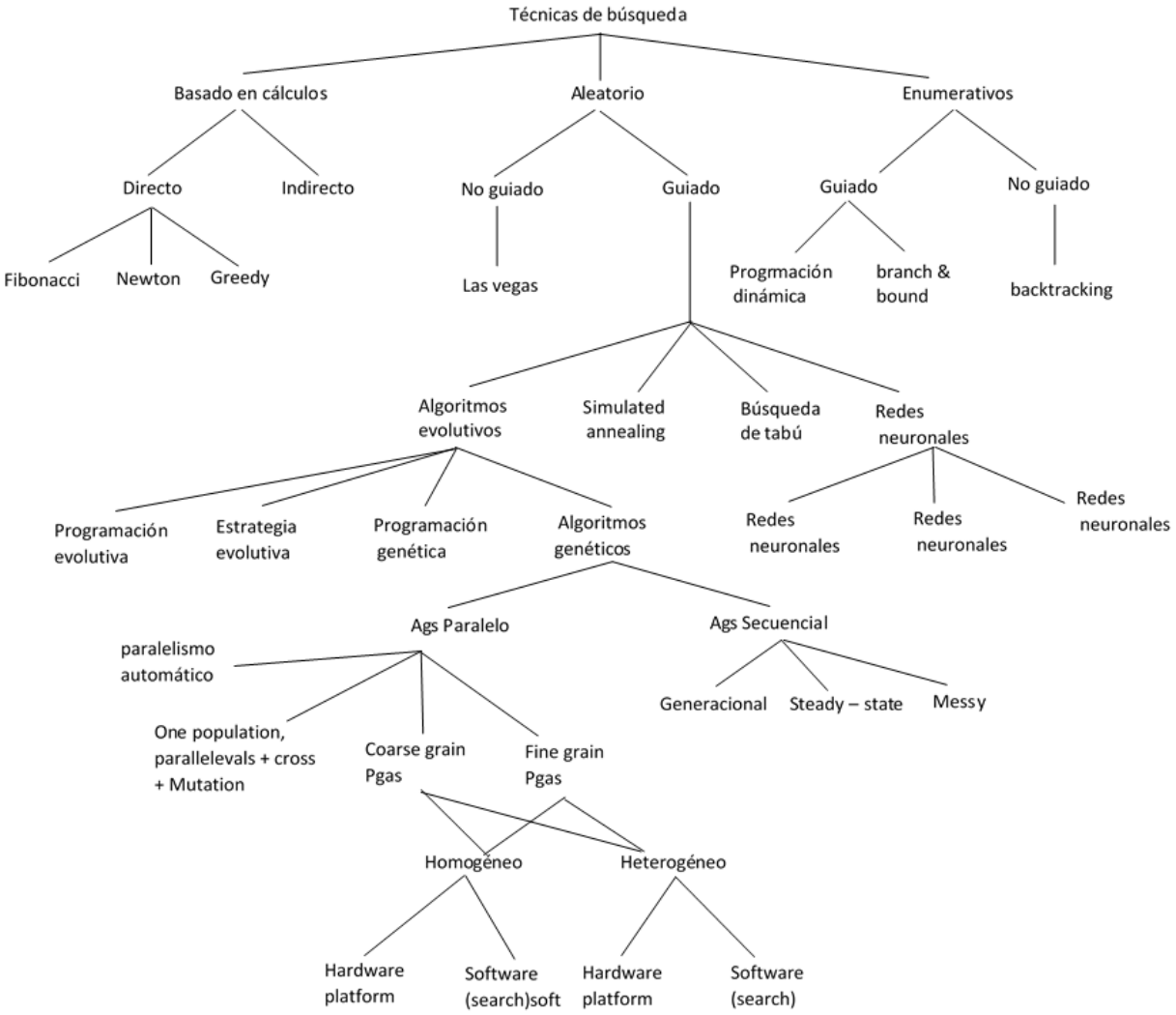


Figura 2.1: Árbol sobre las técnicas de búsqueda
(Sivanandam and Deepa, 2008, pág.IV)

Los AGs se han abierto paso como una poderosa herramienta para resolver problemas de búsqueda y optimización, puesto que tienen una robustez en espacios complejos y computacionalmente son simples pero potentes. Existen terminologías que es menester saber para poder comprender el mecanismo de los AGs. Sin embargo, como utilizan palabras ya definidas en la biología, se necesita entender puesto que se emplean para redefinir la aplicación de los mimos los cuales se verán más adelante.

2.2. Contexto genético

En principio, la genética se define como la rama de la biología que se encarga del estudio de la herencia y las variaciones que existen en los organismos. La herencia es la transmisión de contenido genético, es decir ADN celular de un ser vivo a sus descendientes. A su vez las variaciones son aquellas modificaciones que se pueden dar en los organismos a nivel genético de una población o una especie.

A primera instancia se sabe que todos los organismos vivos están formados por células, las cuales son consideradas como la unidad básica fundamental de cada individuo, son capaces de actuar de manera autónoma. Estas están construidas básicamente por citoplasma, membrana y núcleo. Dentro del núcleo de las células que componen a los seres vivos, se hallan los organelos llamados cromosomas, los cuales son estructuras altamente organizadas y formadas por largas cadenas de moléculas de Ácido Desoxirribonucleico (ADN) asociado al Ácido Ribonucleico (ARN) y proteínas. El ADN es un ácido nucleico, descrito como una escalera en forma de doble hélice, el cual está compuesto por estructuras más simples llamadas bases nitrogenadas, las cuales son 4: Adenina, Guanina, Citosina y Timina. Tiene la principal función del almacenamiento de toda la información genética a largo plazo; esta información sirve para el desarrollo y la construcción de otros componentes de las células como las proteínas y las moléculas de ARN entre otras funciones.

Los segmentos de ADN que están a lo largo de la cadena, que contienen parte de la información, son llamados genes, los cuales contienen instrucciones para dar forma a nuevas moléculas como las proteínas; es decir, codifican las propiedades de los individuos siendo estas sus características como el color de ojos, color de pelo, la forma de la nariz etc. Un importante atributo es que permiten heredar las características a sus descendientes, es aquí que el principal enfoque del estudio de la genética son estos mismos. El conjunto de genes se llama genoma siendo un conjunto de características que tiene el individuo. Los alelos se pueden definir como estados alternativos en un gen; por ejemplo, los colores posibles que pueden tomar los ojos siendo: azules, verdes, marrones, etc, En específico, las variantes que pueden presentar pueden ser por mutación, de este modo los genes toman un conjunto de alelos diferentes. El conjunto de todo el material genético que dictan las características del individuo es llamado genotipo, mientras que la expresión física de todo este material genético codificado es el fenotipo. El locus por su parte es una posición particular en un cromosoma.

En la siguiente imagen (figura 2.2) se pueden ver gráficamente la estructura de una célula de forma simple con los componentes explicados anteriormente:

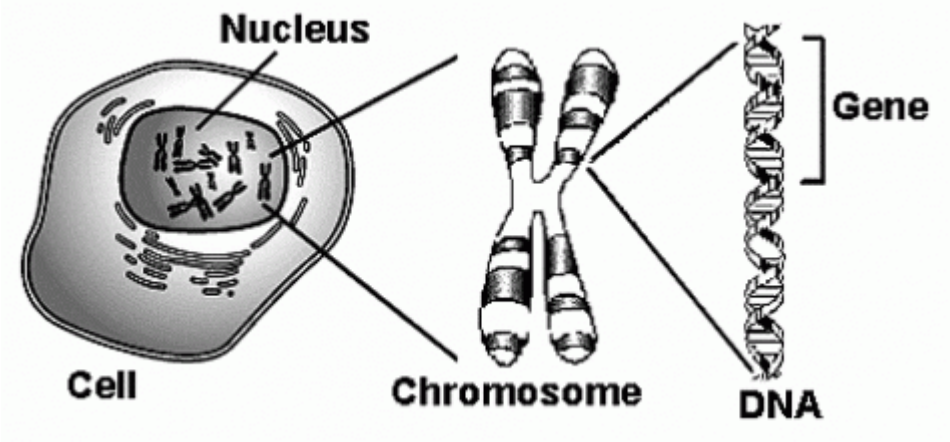


Figura 2.2: Estructura de una célula
(Biology Notes for IGCSE 2014, 2014)

Dentro del contexto de la evolución, la variación genética es una parte vital puesto que permite una mejor adaptación de la especie con respecto al entorno ya que permite que exista mayor diversidad en el grupo dándole más probabilidad de supervivencia ante los cambios de su entorno; de no tenerla, se tendría mayor riesgo y posiblemente implicaría la extinción de la especie. Dentro del tema existen dos procesos moleculares que dan la posibilidad para generar dicha variación, siendo la recombinación y la mutación. Las mutaciones son cualquier cambio o alteración en una secuencia de ADN de un individuo; estas mutaciones pueden ser beneficiosas para el mismo o pueden ser perjudiciales, pero gracias a ellas la especie ha tenido una mejor adaptabilidad a su entorno. Estas pueden ser heredadas o transmitidas a la descendencia, que pueden ocurrir de manera espontánea o inducida por agentes mutagénicos; estos pueden ser físicos, químicos o biológicos. Cuando esto llega a suceder, altera el organismo aumentando las mutaciones por encima del umbral. Algunas mutaciones pueden ocurrir por errores durante la división celular, error en la recopilación del ADN, por exposición a mutágenos como la radiación que puede alterar la secuencia y la composición de los genes siendo de tipo físico, etc.

La recombinación solo ocurre durante la división celular e implica que una hebra de ARN o ADN se divida y se una a una molécula de material genético diferente formando una nueva secuencia de ADN de manera que la molécula de ADN original está presente. Por lo general, esta es la forma en que la descendencia obtiene una combinación diferente a la que tenían los padres. Esta recombinación se da en la meiosis en células eucariotas por ejemplo: esto es importante porque trae a tema la reproducción, la cual es la capacidad que tiene todo ser vivo para tener descendencia ya sean iguales o semejantes a ellos. Actualmente existen dos formas de reproducción asexual y sexual.

La Mitosis es la base de la reproducción asexual, la cual lleva a cabo el proceso de división celular con el objetivo principal de reproducción de una célula hija exactamente igual al padre, generalmente es acompañada de una sola división celular; esto no proporciona ninguna diversidad genética. Por su parte, la reproducción sexual tiene como base a la Meiosis; este es el proceso de división celular el cual parte de la unión de dos células sexuales especiales llamadas gametos que comparten información mediante una cruza de material genético, posteriormente se recombina la información genética derivando en cuatro células hijas con la mitad de cromosomas que tenían originalmente los padres, hablando en términos generales. Este proceso claramente abre paso a la capacidad de evolución de las especies y permite ser una fuente de variabilidad genética. En la figura 2.3 se muestra las dos formas de reproducción tanto de la mitosis como la meiosis.

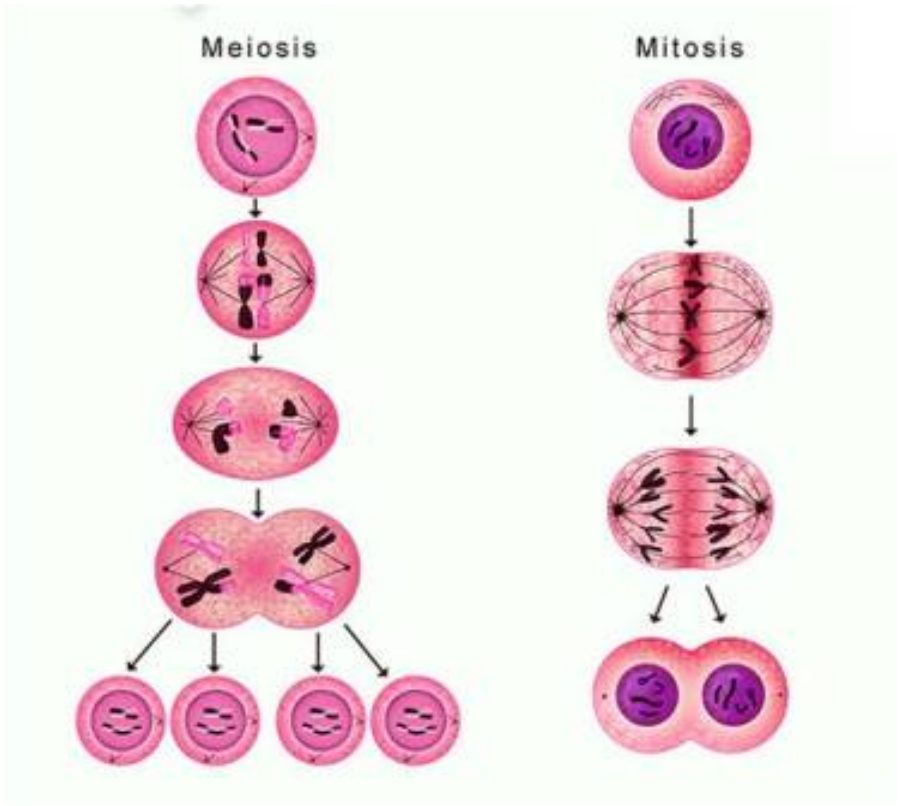


Figura 2.3: Forma de reproducción de la Meiosis y la Mitosis
(Cuba Educa, 2014)

Las diversas terminologías biológicas tratadas anteriormente son la base de la construcción de los AGs. Es por ello que es importante entender estos conceptos para comprender el mecanismo de los mismos. En la siguiente tabla 2.1 se presentan algunos términos comunes entre la evolución natural y el AG.

Evolución Natural	Algoritmo Genético
Cromosoma	Cadena (Individuo)
Gen	Características
Alelo	Valores de las características
Locus	Posición en la cadena
Genotipo	Cadena codificada
Fenotipo	Conjunto de parámetros

Tabla 2.1: Comparación entre AG y selección natural
(Sivanandam and Deepa, 2008, pág.20)

2.3. Composición básica de un Algoritmo Genético

2.3.1. Cromosomas

Toda la información que compone a un individuo está contenida en el cromosoma. La representación más común es una longitud fija de bits; es decir una cadena en un AG simple que se maneja tradicionalmente, como se muestra en la siguiente imagen (figura 2.4), cada dígito que compone la cadena de ceros y unos es llamado alelo que representa alguna característica o valor del individuo que es contenida en un gen. A su vez, un gen es un fragmento mínimo de cadena que está en el cromosoma, genotipo es la combinación completa de genes conteniendo todas las características del individuo, mientras que el fenotipo es el resultado de ello representado en una estructura derivando en un conjunto de parámetros.

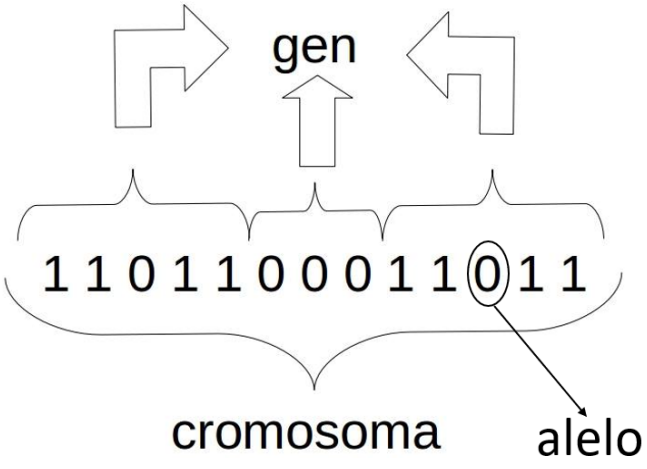


Figura 2.4: Cromosoma binario
(Elaboración propia)

2.3.2. Función objetivo

Esta función también es conocida como función de adaptación o de evaluación, es muy importante definirla ya que esta función nos permite conocer la aptitud de cada individuo, puesto que es el valor de la función mediante el cual podremos valorar el mejor entre toda una población. En algunos casos los individuos no están en términos en los que se pueda evaluar directamente en la función, así que primero se tiene que decodificar para poder proceder con la evaluación y esto tiende a suceder principalmente con los individuos que están en términos binarios. La información que nos puede dar este valor es un indicador de que tan cerca está del óptimo global; sin embargo, hay ocasiones en los que no se conoce el óptimo global y se puede confundir con un óptimo local puesto que está lejos en el espacio de búsqueda y no se puede avanzar fácilmente; en este caso la mutación puede ser una buena ayuda. Se debe contemplar las restricciones, condiciones o penalizaciones para poder considerar un valor una buena solución dependiendo del tipo de problema a tratar y el tipo de solución que se contemple en un caso particular.

Un contratiempo que sucede en la ejecución de los AGs se relaciona con la velocidad en la que converge el algoritmo. Cuando la convergencia es rápida se denomina prematura, en este caso se converge en óptimos locales, los ya mencionados. El problema radica cuando este óptimo local aparece de forma precipitada y define el comportamiento de tal forma que sea difícil salir de este punto en el área de búsqueda. En este caso se deberían contemplar medidas para que estos no dominen a la población. Por otro lado, un problema que se tiene es precisamente lo contrario, que el tiempo de convergencia sea muy largo, sus inconvenientes es evidentemente el tiempo en el que converge.

2.3.3. Población

El conjunto de toda la información que compone a uno o hasta n cromosomas, dependiendo como se estipule, es identificado como individuo y a su vez un conjunto de estos individuos es llamado población. Las generaciones se pueden identificar como la descendencia que se obtenga a partir de una población anterior creando una nueva siendo esta la actual, es importante estipular el tamaño de la población inicial así como el número de generaciones que se crean, de igual manera es importante determinar el tamaño de los cromosomas de los individuos; el tamaño de la población será determinada por la complejidad del problema y generalmente la población inicial se crea aleatoriamente siendo cada individuo una posible solución en un espacio de búsqueda amplio, o se puede iniciar con soluciones ya conocidas o con métodos heurísticos. Con esta estrategia se iniciaría con una aptitud alta y eso le ayudaría al AG a encontrar la solución ideal pronto.

Es importante que haya diversidad entre la población pues entre mayor sea más región de búsqueda será explorada, de lo contrario si hay poca solo se limitará a cubrir una pequeña región y no podrá encontrar el óptimo global y esto, en parte, es definido por el tamaño de la población. Sin embargo, entre más población sea estipulado mayor será el costo computacional, memoria y tiempo.

2.3.4. Operadores Genéticos

Selección

La selección es un proceso por el cual se eligen a dos individuos de la población para crear a un tercero y formar a la nueva descendencia. El objetivo de la selección tiende a elegir con mayor probabilidad a los individuos más aptos de entre todos los que hayan sobrevivido con la esperanza de que las mejores características sean heredadas para lograr que los hijos tengan una mejor aptitud. En esta parte uno puede darse cuenta que la base de la selección es la teoría de Darwin donde los mejores tienen más probabilidad de sobrevivir y con ello de tener descendencia. Lo interesante es el proceso mediante el cual se eligen estos individuos para producir a la nueva generación, ante todo se debe estipular el número de nuevos individuos que se formarán por cada par y el número o tamaño de los cromosomas.

Lo que se espera idealmente de un método de selección es encontrar una forma mediante la cual la presión de selección y la diversidad de la población se puedan ajustar al rendimiento de la búsqueda, ésto es muy importante, puesto que determina la estrategia para encontrar el óptimo global. Existen varios métodos para este proceso, sin embargo, tradicionalmente se hace mediante la selección proporcional a la función objetivo, también llamado selección por ruleta.

Selección proporcional

La selección proporcional es un proceso aleatorio donde se le proporciona un porcentaje a cada individuo de la población para que sea seleccionado, siendo un número proporcional a su aptitud, de tal forma que la suma de todos los porcentajes formen la unidad. Este porcentaje representa una porción de una ruleta que simulará una rotación, cuando se detenga se sabrá el individuo elegido dependiendo en que parte de la ruleta cayó la aguja. La aptitud de cada individuo influye al momento de proporcionar el tamaño dando una proporción mayor a aquellos individuos mejor dotados de forma que tendrá más probabilidad de ser elegido y en contraparte el individuo con menos aptitud tendrá menos posibilidades de ser elegido.

Dado que x_i representa el cromosoma del individuo i, $f(x_i)$ es el valor asignado de la función objetivo y p_i la probabilidad de que un individuo sea seleccionado como padre, se tiene que:

$$p_i = \frac{f(x_i)}{\sum_{i=0}^{n} f(x_i)} \text{ donde } i = 0, 1, 2, ..., n$$

Un problema que se presenta si es que se selecciona este método es que, los súper individuos son beneficiados de mayor manera tomando control de la población. Esto provoca que la diversidad sea menor, con ello se favorece a la rápida convergencia; esto implica que se presente un problema ya explicado anteriormente. De igual manera, si la selección es muy débil, la convergencia será lenta dando como resultado una evolución demasiada rezagada. En este contexto se debe equilibrar la selección y para ello se pueden utilizar algunas mejoras del método de mutación, variación del cruce entre otras.

Selección por torneo

La selección por torneo consiste en fijar una cantidad de n individuos, estos mismos participan en una competencia para determinar quién es el mejor dentro del grupo actual. El proceso es llevado a cabo para elegir a los futuros progenitores; este procedimiento es repetido hasta tener la cantidad deseada de padres que pasarán a la siguiente fase del ciclo del AG. Este método tiene variantes al implementarse y se puede dividir en dos caminos principales, determinista y probabilística:

- En la versión determinística se toma un número aleatorio p, que representa la cantidad de individuos que formará cada grupo para competir, de entre ellos solamente pasará a la siguiente etapa, aquel que tenga un mayor valor de adaptación.
- La versión probabilística difiere de la determinística en el proceso para determinar el individuo que pasará al grupo de progenitores. En vez de escoger al mejor en todos los casos, se genera un número aleatorio r entre el intervalo $[0, 1]$; si r toma un valor superior al umbral, especificado como un parámetro del algoritmo, el mejor individuo pasa a formar parte de la reproducción, en caso contrario se toma al peor, siempre bajo el criterio del valor arrojado por la función objetivo para la evaluación del peor o mejor.

Estos dos son los métodos más conocidos de entre la selección, pero existen más métodos y cada uno de ellos tienen más variaciones para su mejora. Cada método tiene sus pros y contras. Por ejemplo, el método de la ruleta tiende a tener una presión de selección elevada, el resultado de ello es una poca diversidad; por otro lado, la selección por torneo reduce ese problema teniendo una presión de selección más baja y permite más diversidad para no estancarse en óptimos globales.

Pese a ello, si no se implementa bien se puede incurrir en una presión elevada, es por ello que dependiendo a las necesidades del problema se elige un método y se adapta para mejorar la búsqueda en el AG.

Elitismo

Un importante operador es el elitismo también conocido como la selección elitista el cual toma una muestra de tamaño fijo, que representa un porcentaje de la población. Este contiene a los mejores individuos de entre toda la población, de esta forma se busca preservar en cada generación a los mejores prospectos sin temor a perderlos en alguna generación ya sea porque no fueron seleccionados o modificados por la mutación o el cruce. Con ello se mejora el rendimiento de manera significativa en el AG.

Cruza

Una vez cumplido el proceso de selección se ha formado un grupo para formar parte de la reproducción que se contemplan en esta fase; esta misma procede a la creación de nuevos individuos. Entre la nueva generación se encuentran prospectos que no existían en la población anterior, la existencia de estos individuos permite expandir el campo de búsqueda habilitando nuevas regiones en el espacio lo cual determina la evolución de la población con la esperanza de que siempre se obtenga un resultado mejor que el anterior y así acercarse al óptimo global. Existen variaciones diversas para realizar esta fase, aunque este proceso tradicionalmente se puede resumir en tres simples pasos:

- El operador de cruce selecciona a dos individuos de forma aleatoria comúnmente, del pool de individuos generados por la selección.
- A lo largo del cuerpo del individuo se selecciona un punto o varios para marcar el sitio de cruza.
- Finalmente, la recombinación procede a realizarse entre los dos padres que intercambiarán información genética siguiendo los sitios o sitio de cruce; esto da forma a una nueva descendencia.

Monopunto

Considerado como cruce básico o tradicional, consiste en la cruza a partir de un único punto el cual parte la cadena en dos nombradas parte posterior y parte anterior. La parte anterior del padre es mezclada con la parte posterior de la madre y la parte posterior del padre con la anterior de la madre para conformar a dos hijos (figura 2.5). Si se elige un sitio apto se obtendrán hijos con mejores aptitudes.

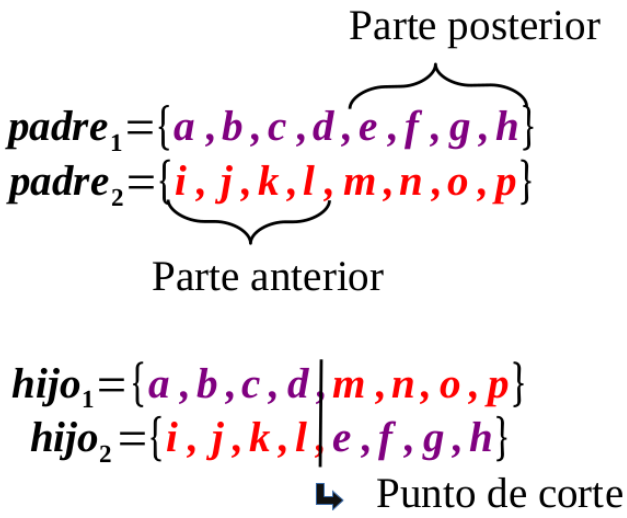


Figura 2.5: Cruce monopunto
(Elaboración propia)

Multipunto

Este método cuenta como base la mecánica del monopunto, la diferencia principal es la cantidad de puntos de corte que se tienen para la cruza. Se establece un número k de forma aleatoria que representa el número de puntos de corte, y de la misma manera que el proceso anterior se procede a la recombinación de los genes del padre con los de la madre para crear a la nueva generación entre los segmentos generados por el corte.

Existen varios procedimientos para realizar la reproducción, así como, variaciones de los mismos, estos se elegirán conforme se adecuen al problema a tratar de forma que puedan beneficiar a la solución del problema.

Mutación

Comúnmente la mutación es un proceso que se implementa después de la cruza sobre la nueva descendencia. Este procede alterando la información del individuo de forma que provea diversidad a la población y es un mecanismo que evita que el algoritmo quede atrapado en un óptimo local. Este operador es considerado un operador de búsqueda que ayuda a alcanzar las regiones que no le es posible alcanzar al algoritmo fácilmente. Existen varias formas de realizar la mutación, por ejemplo (figura 2.6): Generalmente al trabajar con cromosomas en forma de cadena binaria, se puede tener un alelo con un valor 1 y procede a tener un valor de 0 y cuando se tiene un 0 se procede a reemplazarlo con 1.

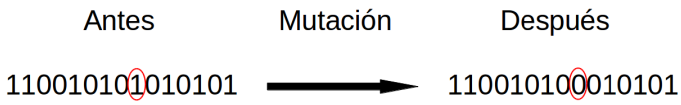


Figura 2.6: Mutación
(Elaboración propia)

El operador de mutación tiene asignado una tasa predeterminada que se aplica para cada individuo, de ser mayor al umbral que se estableció al principio se procede a aplicar el operador y de no superar este umbral simplemente se pasa al próximo individuo para ver si se implementa o no hasta que termina de recorrer a la población. Habitualmente la tasa para el operador es bastante pequeña con un 0.1 % normalmente, pero de igual manera se ajusta de forma que convenga al problema.

Siempre se debe considerar que existe la posibilidad que la mutación lleve a que un individuo tenga un peor resultado después de ser mutado ya que puede romper las posibles correlaciones entre genes que se formaron, pero los resultados obtenidos por la mutación lo hace muy fiable cuando la implementación es adecuada.

2.3.5. Remplazo

El reemplazo es la última etapa, en la mayoría de los casos los AGs mantienen un tamaño poblacional predefinido desde la primera generación aunque también hay variaciones. El reemplazo no es otra cosa que la integración de la nueva descendencia a la población nombrándola nueva generación. Sin embargo, existen diferentes modos de emplear esta parte, los más conocidos son los siguientes:

- Generacional: El eje principal es crear N individuos de un tamaño poblacional N, esto conlleva a reemplazar a todos los individuos que existen en el conjunto. Esto implica que la reproducción solo se pueda realizar entre prospectos de la misma generación.
- Estado estacionario: A comparación del reemplazo generacional se genera un número predeterminado de descendencia que posteriormente se incluirán para formar parte de la nueva generación. Este proceso generalmente requiere reemplazo de algunos miembros de la generación anterior, lo que significa que se conserva parte de la población de la generación anterior.

Existen varios criterios para establecer en los Algoritmos Genéticos que se pueden implementar tanto al método generacional como al método de estado estacionario, se puede elegir entre, sustituir completamente a los padres por sus hijos, elegir a los individuos que se sustituirán de forma aleatoria. El número a sustituir está dado por el tamaño de la descendencia, por reemplazo de los individuos peor adaptados o por reemplazar por individuos de adaptación similar.

Este proceso toma a cada individuo en la nueva generación y lo evalúa con la función de aptitud que se ha establecido previamente, que corresponde al problema a resolver. En este punto se podrá ver que la evaluación depende totalmente de la calidad del individuo, puesto que entre más calidad mejor aptitud tendrá. En algunos casos, en esta parte a los individuos mejores adaptados se les premia y consecuentemente se les castiga a los que tienen peor resultado. Esto se hace con el fin de que se propaguen los mejores genes y controlar a los peores limitándolos.

2.3.6. Términos de búsqueda

Es menester establecer un criterio o criterios en donde el AG deja de evolucionar, de esa forma poder terminar la búsqueda. La condición de terminación es determinada por uno mismo de acuerdo al objetivo que se tenga para la solución del problema. Lo más común se resume en: estipular un número de generaciones, establecer un lapso de tiempo o no percibir un cambio en el resultado de la función de adaptación. Existen más reglas que se pueden implementar y variaciones con las ya mencionadas.

Finalmente, los AGs explotan la información histórica para conjeturar un nuevo punto de búsqueda con un mejor rendimiento, de esa forma exploran nuevas regiones y encuentran nuevos puntos que se acercaran cada vez más al óptimo global. El punto clave del tema es tener una solidez del equilibrio entre la eficiencia y la eficacia necesaria para la supervivencia de cada uno de los individuos.

Capítulo 3

Sistemas Caóticos

3.1. Teoría del caos

Ante todo, es imperativo entender la teoría del caos puesto que es una base fundamental de los sistemas caóticos, partimos de la definición de la Real Academia Española (RAE) de caos, que la define como: "El comportamiento aparentemente errático e impredecible de algunos sistemas dinámicos deterministas con gran sensibilidad a las condiciones iniciales".

Esto es un resumen de lo que en verdad comprende esta teoría. A primera vista se puede definir con las palabras desorden y confusión en el lenguaje común pero más allá de esto se hace referencia a un caos determinista, como menciona la definición anterior, del cual se hablará más adelante. Se descubrió desde el punto de vista de varias disciplinas a partir de los 60's. Todo comenzó con un meteorólogo llamado Edward Lorentz el cual utilizó un modelo matemático de 12 ecuaciones no lineales para predecir el tiempo, esto le ayudaba a crear una simulación probable del comportamiento de la atmósfera en su computadora. Lorentz utilizaba 6 decimales para dicha simulación, pero cuando volvió a correr la simulación para ahorra tiempo y papel solo introdujo los datos con tres décimas, lo que nunca imaginó fue el resultado que obtuvo, puesto que el resultado era totalmente diferente al que había arrojado la computadora anteriormente y con ello nace una nueva teoría(Gleick, 2011, pág.16). Actualmente la teoría del caos es la rama de muchas ciencias como la Biología, las Matemáticas, la Física, la Economía, etc. Nace de la necesidad de entender la complejidad de la naturaleza desde el punto de vista de varias ciencias, esto ayuda a detectar un orden, un patrón donde existen en cierto tipo de sistemas con comportamientos erráticos y aleatorios; estos son complejos y dinámicos no lineales. Sin embargo, los sistemas que estudia esta teoría son deterministas.

Antes de continuar es menester repasar algunos conceptos antes de dar una definición de lo que es un sistema caótico.

3.2. Sistema

Un sistema es una combinación de componentes que actúan conjuntamente para alcanzar un objetivo específico. Una componente es una unidad particular en su función en un sistema. De ninguna manera limitado a los sistemas físicos, el concepto de sistema se puede ampliar a fenómenos dinámicos abstractos, tales como los que se encuentran en la economía, el transporte, el crecimiento de la población y la biología (Ogata and Sanchez, 1987, pág.1).

3.2.1. Sistemas dinámicos

Un sistema dinámico describe cada uno de los puntos a lo largo del camino en un espacio dado, esto quiere decir que se define por aquella dependencia que existen entre la salida actual y las variables anteriores e iniciales, con ello describe los estados futuros que siguen del estado actual evolucionando en el tiempo. En palabras comunes se puede definir como causa y efecto, se podría ver de la siguiente manera: los efectos actuales serían las salidas que son el resultado de causas actuales y previas representadas como entradas de acuerdo con la definición anterior. Como se puede ver se podría decir que estudia sistemas deterministas.

Tres elementos se consideran esenciales para estar presentas dentro de este tipo de sistema:

- Parámetros: Características que no cambian con el tiempo.
- Variables: Elementos que cambian con el tiempo.
- Leyes: Principio que cambia con el tiempo.

3.2.2. Sistemas discretos

Un sistema discreto es aquel que cambia instantáneamente de valor en ciertos instantes de tiempo, es decir, presentan discontinuidad tanto en el tiempo como en la magnitud, es por ello que el tiempo es medido en pequeños lapsos. En estos sistemas se toma el estado actual como entrada y se actualiza la situación con cada estado de salida que se convertirá en la entrada para obtener el siguiente y así sucesivamente.

3.2.3. Sistemas deterministas

Un sistema determinista se caracteriza porque en el desarrollo de sus estados en el tiempo no se involucra el azar, es decir, a partir de un estado inicial o partiendo de condiciones iniciales predeterminadas siempre se obtendrá la misma salida.

3.2.4. Sistemas complejos

Este tipo de sistema se caracteriza principalmente por ser imprevisible por su difícil comprensión. Como se ha mencionado, no hay una definición en concreto de este tipo de sistemas pero un conjunto de características son englobadas constantemente dentro de su descripción. Como se menciono anteriormente lo que es un sistema, este no es tan simple de analizar puesto que las partes entrelazadas forman vínculos que arrojan información extra que no es fácil de observar. Quiere decir que el estudio de estas partes de forma aislada no le da al observador una vista clara del sistema en general ya que su comportamiento es distinto al estar interconectados a otros elementos. Con ello se quiere decir que no basta el solo analizar la función de cada uno de sus elementos, sino analizar también el funcionamiento del sistema una vez que estén conectados todos sus elementos, saber sus estados actuales de cada uno y sus transiciones ya que existen variables que están ocultas a simple vista y se hace difícil observar el sistema con precisión; en palabras más comunes saber muy bien su estructura y función.

3.2.5. Sistemas no lineales

Los sistemas no lineales se representan cuando al menos una de sus ecuaciones no es lineal, es decir, que por lo menos una de sus incógnitas es mayor a primer grado. De esta forma el comportamiento del sistema no se expresa como la suma del comportamiento de sus descriptores. Un dato interesante es que en diversos campos la no linealidad es causa de comportamientos complejos ya que el sistema no opera bajo el principio de superposición.El cambio de la salida no es proporcional al cambio de la entrada, esto quiere decir que puede responder de una forma muy diferente ante la misma entrada con condiciones iniciales diferentes.

Un sistema no lineal tiene la siguiente forma:

$$F(u) = 0$$

Para algún valor desconocido de u.

3.2.6. Sistemas estables

Un sistema estable es aquel en el cual sus propiedades y operaciones no sufren cambios importantes o lo hacen solo en ciclos repetitivos independientemente de las variaciones, quiere decir que el resultado de una salida limitada es debido a una entrada limitada estipulada. Este tiende a través del tiempo a un punto u órbita.

3.2.7. Sistemas inestables

Un sistema inestable no siempre es constante, de hecho muestra un comportamiento errático, se representa cuando una ligera perturbación o un pequeño cambio en la entrada tiende a causar grandes repercusiones en su salida. Por el contrario de los sistemas estables, la salida no está limitada en el sistema por la entrada limitada y se puede decir que puede escapar de los atractores.

3.3. Puntos de equilibrio

Estos puntos en sistemas no lineales se encuentran en el momento en el que se igualan a cero sus ecuaciones.

$$x = 0 \rightarrow f(x) = 0$$

En el sistema no lineal se puede tener más de un punto de equilibrio mientras que en sistemas lineales solo puede tener un único punto de equilibrio. Una característica importante es que una vez el sistema se halla en este punto, se queda en el mientras no exista un factor externo que lo afecte. El nombre de puntos de equilibrio se halla en sistemas continuos, mientras que en sistemas discretos se denominan puntos fijos.

3.4. Atractor

Una definición de forma rigurosa es difícil obtener, sin embargo, puede tomarse de forma general cuando el comportamiento de un sistema tiende a un conjunto de valores numéricos, siendo un conjunto cerrado, independientemente de las condiciones iniciales en el sistema.

Quiere decir que es una descripción de la rutina que tomará un sistema a lo largo del tiempo que siempre acaba teniendo. Se dice que es un comportamiento que al sistema le atrae mucho tener; este posee las siguientes propiedades:

- Es invariante frente a la dinámica del sistema: Cualquier trayectoria que comience en el atractor permanece en él indefinidamente.
- Atrae a un conjunto abierto de condiciones iniciales suficientemente cercanas a él. Al conjunto más grande de tales condiciones se le denomina cuenca de atracción.
- Es mínimo: Ningún subconjunto de él puede satisfacer las propiedades anteriores.

3.5. Definición

No existe una definición completamente concreta sobre el tema, pero se comprende como un sistema complejo, dinámico, no lineal, determinista que evoluciona a través del espacio de fase que parece ser bastante aleatorio. Como se mencionó anteriormente, es un sistema estable e inestable, lo cual se presenta en este tipo de sistema al tender a manifestar los dos comportamientos. Una fuerte singularidad que caracteriza fuertemente a los sistemas caóticos es que tienen una sensibilidad alta a las condiciones iniciales; esto significa que un pequeño cambio en sus condiciones iniciales se amplifica y extiende exponencialmente a lo largo del sistema desencadenando un comportamiento totalmente distinto, esto lleva a ser imposible una predicción del estado final de un sistema caótico sin las condiciones iníciales correctas. Esto es porque el sistema, pese a aparentar ser aleatorio, no lo es, sino que es totalmente determinista pues solo en apariencia puede aparentar estar en desorden ya que tiene un orden interno. Pese a lo mencionado anteriormente sobre una definición no clara, existen características que se usan para describir a un sistema tipo caótico. Estos son los siguientes:

- Extrema sensibilidad a las condiciones iníciales
- Proceso pseudoaleatorio
- Determinismo
- No-linealidad
- Ergodicidad y confusión
- Dependiente de la información histórica
- Periodicidad infinita

3.6. Teoría de la bifurcación

Todo lo que comprende la teoría de la bifurcacion es englobado en un campo matemático donde el eje principal es sobre el estudio del cambio cualitativo en la estructura usualmente en sistemas dinámicos. Una bifurcación surge comúnmente cuando al variar uno o más parámetros con un pequeño cambio en sus valores de un sistema, provoca un brusco cambio repentino cualitativo en su comportamiento. El valor que causa esta conducta es nombrado valor de bifurcación o valor crítico.

Una forma de visualizar de forma global una bifurcación de una ecuación, es mediante un diagrama de bifurcación que es un gráfico que representa las posibles soluciones, es decir, los puntos fijos y periódicos del sistema en función del valor del parámetro de control del sistema, este permite conocer el comportamiento del sistema.

3.7. Aplicación logística

La aplicación logística también llamada mapa logístico es un sistema dinámico discreto que se representa por una ecuación diferencial no lineal. Este es un mapeo polinómico de segundo grado, se popularizó en 1976 debido a un artículo del físico Robert May y que Mitchell Feigenbaum estudió detenidamente de forma más profunda. May estudiaba una forma de entender una manera más simple la dinámica poblacional, en pocas palabras, la ecuación describe el valor futuro de la población a partir del valor actual, siendo x_n la fracción de individuos en un territorio tomando a 0 como la representación de ningún individuo en el territorio y 1 la máxima capacidad posible de individuos. Finalmente, r es un parámetro que representa la tasa de mortandad y la reproducción.

Esta ecuación pretende describir dos efectos:

- Reproducción donde la población aumentará en forma proporcional a la población actual cuando el tamaño sea pequeño.
- Inanición donde la tasa de crecimiento disminuirá proporcionalmente a la capacidad de competencia de los individuos entre sí para asegurar el alimento necesario.

Ecuación:

$$x_{(n+1)} = r * x_n(1 - x_n)$$

Donde:

$$x_n \in [0, 1], r \in [0, 4], n \in [0, \infty]$$

Un hecho muy curioso que descubrió May fue que al variar el valor del parámetro de control del sistema, podía observar valores muy distintos entre sí y en ocasiones muy complejas. Pese a que la ecuación sea una expresión muy sencilla, es un ejemplo perfecto de un sistema caótico.

En el diagrama 3.1 se presenta un diagrama de bifurcación del modelo de May donde el eje de abscisas representa los posibles valores del parámetro r, donde al variar su valor se obtiene diferentes modelos de población y el eje de las ordenadas representa los posibles valores que toma en función de r.

La ecuación logística tiene dos puntos fijos, el primero es el origen y el segundo siendo dependiente del parámetro r. Aplicando la condición de punto fijo, donde el punto fijo tiene que ser igual al valor de la coordenada en el punto fijo, se obtiene:

$$f(x_{pf}) = x_{pf} \quad \rightarrow \quad x_{pf} = rx_p f(1 - x_{pf})$$

Resolviendo la ecuación

$$x_{pf} - rx_{pf}(1 - x_{pf}) = 0 \quad \rightarrow \quad x_{pf}(rx_{pf} + 1 - r) = 0$$

Se obtienen los puntos fijos:

$$rx_{pf} + 1 - r = 0 \quad \rightarrow \quad x_{pf1} = 0, \quad x_{pf2} = 1 - \tfrac{1}{r}$$

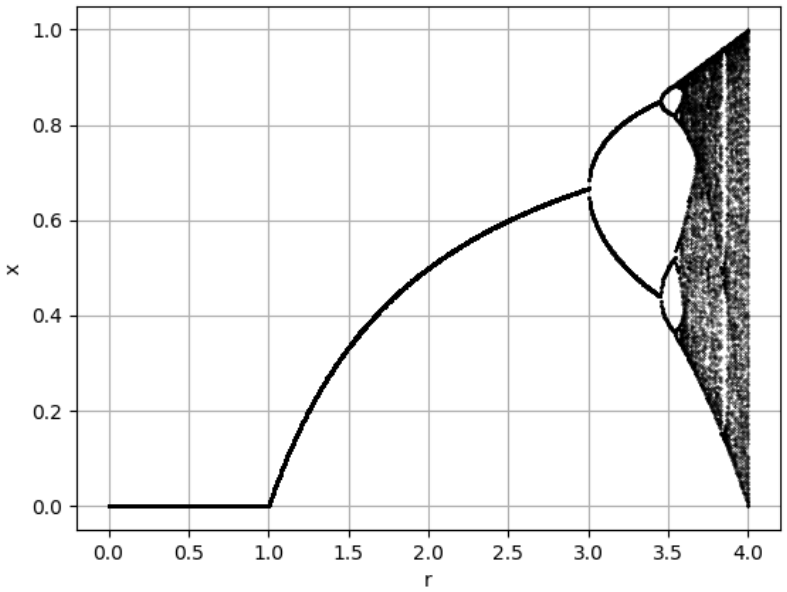


Figura 3.1: Diagrama de bifurcación de la aplicación logística
(Figura propia)

En el diagrama de bifurcación (figura 3.1) se puede ver que cuando $0 < r \leq 1.0$ colapsa a 0 (extinción), para el caso donde $1.0 < r \leq 3.0$ tiende al punto fijo $1 - \frac{1}{r}$, podemos dar un rápido vistazo tomando a $r = 3.0 \quad \rightarrow \quad x = 1 - \frac{1}{3.0} = 0.6666$. Este resultado se puede visualizar en el diagrama y a partir de ahí se puede observar la primera bifurcación que se divide en dos caminos discretos. Al verificar los valores después de esta, sucede lo mismo que cuando $r = 3.2 \quad \rightarrow \quad x = 1 - \frac{1}{3.2} = 0.6875$, el cual no coincide con la gráfica pues para el valor donde $r = 3.2$, x oscila entre dos posibles valores uno entre 0.5 y 0.8. Siguiendo la observación tenemos que, para los valores de $3.0 < r \leq 1.0 + \sqrt{6}$, tenderá a cuatro posibles valores que son las próximas 4 bifurcaciones que se notan entre ese periodo. Para valores donde $3.45 < r \leq 3.54$, los valores oscilarán en cuatro posibles valores de población. Con $r \leq 3.54409$, las bifurcaciones aumentarán hasta que el sistema sea capaz de llegar a cualquier valor ya que oscilará entre 8 posibles valores luego 16, 32 y así sucesivamente. A partir del 3.57 empieza el caos, aunque hay espacios donde presenta un comportamiento no caótico. Para cuando el parámetro llegue a 3.9, tendrá tantos valores posibles por todas las bifurcaciones existentes que aparentará ser un valor al azar pero sabemos perfectamente que no es así. Finalmente, con $r = 4$, los valores dejan el intervalo $[0, 1]$ y divergen para casi todos los valores iniciales.

Esta ecuación es necesaria para nuestra aplicación, la cual se verá en los próximos capítulos y es un ejemplo muy interesante de un sistema caótico.

Capítulo 4

Criptografía

4.1. Definición

Criptografía proviene del griego $\kappa\rho\upsilon\pi\tau\acute{o}\varsigma$ (kryptós): "oculto" y $\gamma\rho\alpha\varphi\acute{\eta}$ (graphé), "grafo" o "escritura", es decir, "escritura oculta". El concepto de la criptografía clásica la define como la ciencia y el estudio de la escritura secreta la cual tiene la tarea fundamental de proporcionar confidencialidad mediante métodos de cifrado. A lo largo de los años este concepto se ha modificado para acoplarse a las necesidades que se requieren, pero para fines de este trabajo tomaremos el siguiente concepto: "Criptografía es la ciencia para crear mecanismos utilizados para cifrar y descifrar cualquier tipo de información utilizando técnicas matemáticas que hagan posible el intercambio de mensajes de manera que puedan cumplir los objetivos del problema planteado".

4.1.1. Los objetivos de la criptografía

Se podría creer que el principal objetivo de la criptografía es la confidencialidad; sin embargo, estos se establecen entorno a las necesidades que se plantean ante un problema que se resuelve mediante este método. De este modo se presentan los siguientes objetivos (El-Samie et al., 2013, pág.4):

- **Confidencialidad**: Se refiere a la protección de la información contra acceso no autorizado.
- **Integridad**: Asegura que la información no ha sido manipulada durante la trasmisión.
- **No repudio**: Significa que el receptor puede probar ante todos que el emisor sin duda alguna envió el mensaje y a su vez este no puede negarlo.

- **Autenticación**: Son clasificados en las siguiente dos categorías:
  - **Autenticación de identidad**: Es el proceso por el cual una parte se asegura de la identidad de una segunda parte involucrada en un protocolo, y que la segunda parte ha participado realmente.
  - **Autenticación de mensaje**: Proporciona autenticación de origen de datos con respecto a la fuente original del mensaje y la integridad de los datos, pero sin garantías de exclusividad y oportunidad.
- **Disponibilidad**: La información debe ser accesible para toda entidad cuando la requiera.

4.2. Criptosistemas

Un criptosistema es un conjunto de procedimientos que se implementan para la seguridad de la información utilizando técnicas criptográficas. Usualmente se compone por tres algoritmos: generador de claves, cifrado y descifrado. Para describir los requerimientos generales que tienen todos los criptosistemas se presentan los siguientes componentes (López, 2011, pág.30):

- M representa el conjunto de mensajes sin cifrar (texto claro).
- C representa el conjunto de textos cifrados (criptograma).
- K representa el conjunto de claves a emplear en el criptosistema.
- E es el conjunto de transformaciones de cifrado (algoritmo de cifrado), que se aplica a cada elemento de M para obtener un elemento de C.
- D es el conjunto de transformaciones de descifrado (algoritmo de descifrado), que se aplica a cada elemento de C para obtener un elemento de M.

Todo criptosistema ha de cumplir la siguiente condición:

$$D_k(E_k(M)) = M$$

En la figura 4.1 se muestra un emisor que tiene un mensaje (texto claro) M que desea enviar, pero para sus fines se debe cifrar y se hace mediante una familia de funciones empleando la clave K_e.

De este modo, se obtiene un elemento cifrado como $C = E_K(M)$ y se envía por medio de un canal público. El emisor recibe este mismo elemento C, con el cual procede a utilizar una familia de funciones (ya conocida por ambas partes) implementando la clave K_d para obtener el elemento original siendo $M = D_k(C)$.

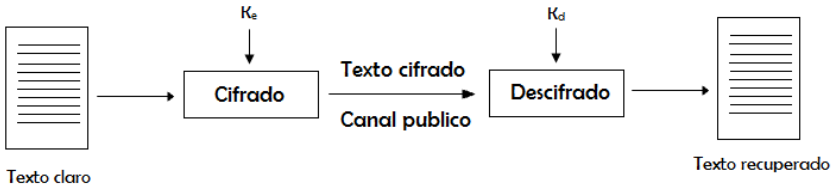


Figura 4.1: Cifrado y descifrado de datos
(Figura propia)

Los criptosistemas deben satisfacer tres requerimientos:

1. Los algoritmos de cifrado y descifrado deben ser eficientes para cualquier llave.
2. El sistema debe ser fácil de usar.
3. La seguridad del sistema debe depender de la llave y no del algoritmo de cifrado o descifrado.

El primer requerimiento es esencial para la aplicación en computadora, los datos son usualmente cifrados y descifrados en tiempo de transmisión, y esta operación no debe crear cuello de botella. El segundo requerimiento implica que debe ser fácil para el usuario encontrar una llave fuerte y finalmente, el tercer requerimiento implica que el algoritmo de cifrado y descifrado sea intrínsecamente fuerte; esto es, que no debe ser posible romper un simple cifrado conociendo el algoritmo usado. Este requerimiento es necesario porque el algoritmo tal vez esté en dominio público o es conocido por el criptoanalista.

4.3. Clasificación de los criptosistemas

Para la clasificación de los criptosistemas se tomará en consideración el tipo de llave, la estructura de cifrado e incluso la cantidad del flujo de datos a cifrar.

4.3.1. Clasificación por tipo de llaves

Existen dos tipos de llaves, también llamadas claves, que se usan comúnmente para cifrar los datos; estos son la llave simétrica y la llave asimétrica.

Las llaves son una parte fundamental de los criptosistemas, puesto que se utilizan para llevar a cabo el proceso para cifrar el texto plano de tal manera que sin ellas no se puede recrear este proceso, de igual manera siendo imposible la recuperación del texto claro por parte de terceros mediante el mismo método.

- **Llave simétrica o clave privada**

  Maneja principalmente el concepto de llave o clave privada, esto quiere decir que esta clave se mantiene en secreto para terceras personas y solo el receptor y el emisor la conocen o al menos es lo ideal.

  Si la relación entre las llaves K_e (llave de cifrado) y K_d (llave de descifrado) es $K_e = K_d$, es decir que ambas llaves son la misma, esto implica que se usa una sola clave al momento de cifrar y descifrar.

  Antes de empezar con el proceso de cifrar y enviar el mensaje, ambas partes debieron haberse puesto de acuerdo con la clave. Usualmente se manda la clave por otro canal distinto relativamente más seguro que el canal por el que se mandará el mensaje. La forma o método de tener esta clave por ambas partes depende mucho de las mismas.

- **Llave asimétrica o llave pública**

  Cuando $K_e \neq K_d$ el cifrado corresponde a un criptosistema de llave asimétrico. En este tipo de criptosistema las entidades manejan dos tipos de llave siendo la llave para cifrar K_e, la llave pública la cual puede estar registrada en un directorio público, es diferente a la llave de descifrado, llamada llave privada K_d la cual es solo conocida por el usuario, donde no es necesario un canal de trasmisión segura para la transferencia de llave.

  Suponiendo que el usuario A (emisor) quiere enviar un mensaje M al usuario B (receptor), A necesitaría saber la llave pública K_e de B para poder cifrarlo de forma que: $C = E_{K_e}(M)$, después de ello lo envía, B lo recibe e implementa su llave K_d para descifrar el mensaje como: $M = D_{K_d}(C)$. Para ello es necesario que ambas partes utilicen el mismo algoritmo de cifrado/descifrado.

  Un aspecto que es importante al crear las llaves que se debe considerar, es que tiene que ser computacionalmente imposible determinar K_d de K_e.

4.3.2. Clasificación por estructura de cifrado

El algoritmo puede ser clasificado acorde a la estructura de cifrado entre cifrado de bloques y cifrado de flujo.

- **Cifrado de bloque**

  Es un tipo de criptosistema simétrico que divide un texto claro M en bloques de longitud binaria fija n, a lo cual utiliza una llave secreta K de longitud binaria r en un algoritmo de cifrado, dando como resultado bloques de texto cifrado de manera que $C = E(K, M)$. Cada bloque de cifrado pasa a tener la misma longitud binaria n. El tamaño común para los bloques son 64 o 128, cuanto mayor sea el tamaño del bloque más seguro será el cifrado pero más complejo será el algoritmo de cifrado/descifrado.

  Los cifrados de bloque pueden ser caracterizados por:

  - Tamaño de bloque: Entre más longitud tenga el tamaño del bloque más seguridad proporciona.
  - Tamaño de llave: Entre más longitud tenga el tamaño de la llave significa que proporciona mayor seguridad.
  - Número de rondas: Rondas múltiples incrementan la seguridad.
  - Modos de cifrado: Estos definen como se cifran los bloques.

- **Cifrado de flujo**

  Es considerado un criptosistema simétrico el cual opera con pequeñas unidades del texto plano M nombrando a estos elementos como caracteres, siendo $m := m_1, m_2, m_3, ..., m_n$ donde $m_i \in M$, con $i \in [0, n]$. Comúnmente se genera una secuencia de bits con una llave K tal que $k := k_1, k_2, k_3, ..., k_n$ donde $k_i \in K$, con $i \in [0, n]$ (llamada flujo de llave), usando un generador de números pseudoaleatorios (Pseudorandom Number Generator (PRNG)) que amplía una llave corta (por ejemplo 128 bits) en una cadena tan grande como el tamaño del texto en claro M (por ejemplo 106 bits). Usualmente se manejan dígitos binarios o bytes, así el cifrado de flujo puede ser diseñado para ser excepcionalmente rápido incluso ante un cifrado de bloque.

  El cifrado se consigue combinando el flujo de claves, con los datos sin formato carácter por carácter de tal manera que para un m_i le corresponde un k_i:

  $$c_i = E(k_i, m_i), i \in [0, n]$$

  Dando como resultado un flujo de cifrado $C := c_1, c_2, c_3, ..., c_n$ donde $c_i \in C$, con $i \in [0, n]$, usualmente se utiliza el operador Exclusive OR (XOR) bit a bit por su simplicidad.

  El cifrado de flujo tiene las siguientes propiedades:

  - No se tiene una seguridad perfecta.
  - La seguridad depende de las propiedades del el PRNG.

- El PRNG debe ser impredecible, dar una consecutiva secuencia de salida de bits, el siguiente bit debe ser indeterminable.
- Típicamente el cifrado de flujo es rápido.

4.3.3. Clasificación acorde al porcentaje de datos a cifrar

Con respecto a esta clasificación, se toma en cuenta el porcentaje de los datos tomados del mensaje original M para crear a C; esto quiere decir que de acuerdo a las necesidades del emisor para el cifrado tiene la opción de hacerlo de forma total (tomando completamente el mensaje M para cifrar) o cifrarlo parcialmente (tomando solo los elementos que son imperativos mantener en privado).

4.4. Criptoanálisis

El criptoanálisis estudia métodos para comprometer la seguridad de un criptosistema, de esta forma ser capaz de obtener acceso al contenido del mensaje cifrado. La rama del conocimiento que incorpora criptografía y criptoanálisis se llama criptología. A su vez, criptología es la disciplina que se dedica al estudio de sistemas, claves y lenguajes ocultos o secretos.

El atacante estudia los detalles de los criptosistemas incluyendo algoritmos y sus implementaciones. Acorde a esto queda claro que la seguridad de un criptosistema debe estar basado en el secreto de la clave, pues se supone que el algoritmo empleado para cifrar es conocido.

Es importante mencionar que los ataques se pueden clasificar dentro de dos grupos:

- Pasivos: El atacante monitorea el canal de comunicaciones tratando de descifrar el mensaje cifrado que recupera en el medio.
- Activos: En este caso el atacante no solo monitorea el canal de comunicaciones si no que manipula el mensaje a su favor.

El mecanismo para obtener los mensajes es indiferente para este punto, lo que interesa es el método de ataque que se emplea una vez obtenido los recursos necesarios. A continuación se presentan algunos de tipo pasivo:

- **Ataque a partir del texto cifrado**: El atacante desconoce el contenido del mensaje por lo cual solo trabaja sobre textos cifrados que ha recuperado en el medio, para tratar de determinar el mensaje en claro.

- **Ataque a partir del texto en claro**: El atacante conoce o puede tener el texto en claro correspondientes a algunas partes del texto cifrado. El objetivo es recuperar el resto del texto en claro usando la información ya obtenida, para ello se trata de determinar la clave o parte de ella.
- **Ataque a partir del texto en claro elegido**: El atacante puede tener acceso a una caja negra, la cual contiene un algoritmo de cifrado con su correspondiente clave que no es conocida, de este modo introducir cualquier texto claro y obtener su cifrado correspondiente, el objetivo es obtener la clave o parte de la misma.
- **Ataque a partir del texto cifrado elegido**: El atacante puede tener acceso a una caja negra que contenga un algoritmo de descifrado con su correspondiente clave que alimenta eligiendo cualquier texto cifrado. El propósito es producir los textos claros correspondientes a los mensajes cifrados que se han introducido para tratar de obtener la clave o parte de la misma analizando los resultados arrojados.
- **Ataque de fuerza bruta**: Principalmente se trata de obtener la clave de forma exhaustiva generando aleatoriamente todos los valores posibles, es por ello que un criptosistema debe ser diseñado de forma que sea computacionalmente imposible deducirla.

4.5. Cifrado de imagen

4.5.1. Cifrado mediante mapa logístico

El mapa logístico que se analizó en el capítulo 3, se utiliza como medio de cifrado para la imagen. En este caso, el parámetro r y el valor inicial (x_0) se consideran clave secreta puesto que se necesitan para poder generar toda la secuencia de dígitos que se utilizarán; por lo tanto, se puede ver que este método maneja llave simétrica.

Veremos un método muy interesante, solo contemplando la implementación del sistema caótico, que ha sido propuesto por Rasul Enayatifar y Abdul Hanan Abdullah (Enayatifar and Abdullah, 2011). Este indica que una imagen debe ser partida en 4 partes iguales y en cada parte se procede a implementar el mapa logístico en cada pixel con una llave diferente, a continuación se presentan los pasos:

1. De cada parte de la imagen que ha sido dividida se extraen los primeros 5 pixeles del número n de la fila que corresponda a la parte que se trabaja ($n \in [1, 4]$), de tal forma que si la parte es $n = 1$, la fila de la que se tomarán los pixeles será la primera, si la parte es $n = 2$ entonces la fila seleccionada será la segunda y así sucesivamente. Estos pixeles ayudarán a crear la llave de cifrado, que servirá para cifrar esa misma parte. De igual, manera formará el valor inicial para la ecuación.

2. El valor inicial de la ecuación está determinado por los siguientes pasos, siendo que:

$$P = [p_1, p_2, p_3, p_4, p_5](\text{Decimal})$$

En la cual p_i representa un bloque de 8 bits, esto surge de esta manera puesto que el valor máximo de un pixel es de 255 y en binario es representado como 11111111 en una escala de grises. P es usado para ser convertido en un número ASCII y esto se pueden ver como:

$$B = [p_{1,1}, p_{1,2}, p_{1,3}, ..., p_{2,1}, p_{2,2}, ..., p_{5,8}](\text{Decimal})$$

Después de que P se convierta en un número ASCII, se genera la cadena B con una longitud de 40 bits, donde ahora P representa el bit j del bloque i-ésimo. Finalmente se obtiene el valor inicial (llave) para comenzar con la ejecución de la función caótica con la siguiente ecuación.

$$U_{0k} = \frac{(p_{1,1}*2^{39}+p_{1,2}*2^{38}+p_{1,3}*2^{37}+...+p_{2,1}*2^{31}+p_{2,2}*2^{30}+...+p_{5,8}*2^{0})}{2^{40}} \text{ donde } k = 1, 2, 3, 4$$

Donde k es el número de cada parte de la imagen, por lo cual al terminar habrá 4 llaves diferentes que se mandarán al receptor, puesto que este paso se repite por cada parte.

3. Una vez teniendo el valor inicial se procede al cifrado de cada pixel. Para empezarse aplica el operador XOR entre viejoValor, es decir el valor del pixel actual y U_{ik} representa al valor i-ésimo de la función caótica en la parte k de la imagen original. A su vez nuevoValor es el que sustituirá al pixel con el valor original convirtiéndose en el actual.

$$nuevoValor = round(U_{ik} * 255) \bigoplus viejoValor$$

Cada uno de los pixeles se cifrará exceptuando los 5 pixeles usados para la llave. Estos pasos son repetidos en cada cuadrante.

Capítulo 5

Algoritmo Propuesto

Una vista general de cómo se operaría con el algoritmo que se propone se puede visualizar en la figura 5.1, el cual presenta el esquema del sistema que se plantea. En el podemos ver que se inicia con el emisor cifrando la imagen, en este caso la de Lenna, con cualquier método para ello. Si el emisor quiere que la imagen tenga una menor correlación entre pixeles o que no tenga vestigios que visualmente deja el cifrado en ocasiones, es decir, que no se logre visualizar algunas características de la imagen original en la imagen cifrada, entonces se utiliza el algoritmo propuesto que generará una imagen nueva a partir de la imagen cifrada y junto con ella un archivo con información para el orden de los pixeles. De este modo, se envía por un canal público los dos archivos, el receptor ya habiendo tenido los parámetros y la llave que el emisor le hizo llegar por un canal privado, solo procede a descifrar y ordenar para ver que la imagen que se le envió fue la de Lenna.

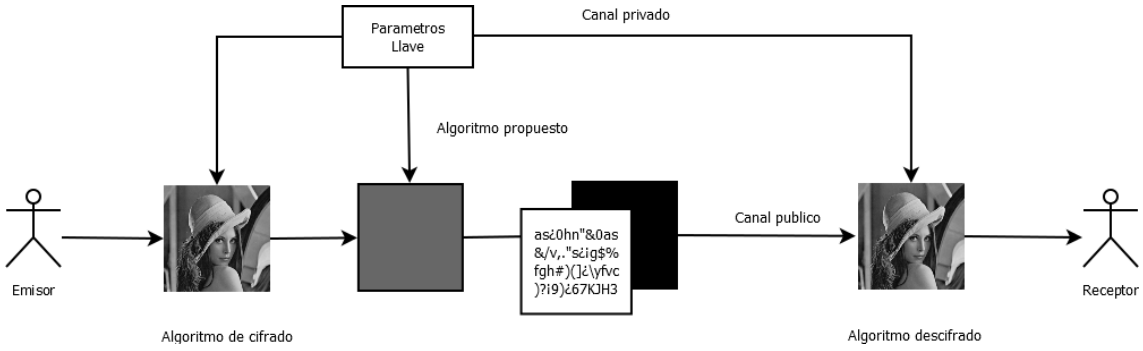


Figura 5.1: Esquema del sistema propuesto
(Figura propia)

El método que se propone para la optimización del cifrado de imágenes se compone de dos fases, cada una de ellas es imprescindible para la función de este.

La primera fase es la creación de un Algoritmo Genético, que en el capítulo 2 se describió, así como las partes que conforman su función. Como bien es sabido, estas partes esenciales son: cromosomas, función objetivo, la población, operadores genéticos (muta, cruza y elitismo). Para la segunda fase se toma en cuenta la generación de un archivo que se cifra por medio de una llave que ayudará en el orden de pixeles de la imagen. El eje del método se lleva a cabo con llave simétrica, del cual se habla en el capítulo 4. Es muy importante este punto puesto que, sin el proceso, el receptor no podría descifrar la imagen y con ello no podría ver el mensaje. Un detalle importante a mencionar es que el archivo es generado mediante aplicación logística del cual se habla en el capítulo 3, es elegida por sus altas propiedades que posee, por ser un sistema caótico pese a ser una función muy sencilla. Estas fases se detallan en los siguientes temas.

5.1. Primera fase

5.1.1. Estructura del algoritmo propuesto

Cromosomas

Los cromosomas son representados como el conjunto de valores de una imagen; ésto quiere decir que es una matriz que contiene el valor de cada pixel, siendo cada pixel un alelo de un rango entre 0 a 255. Cada cromosoma corresponde a cada fila de dicha matriz, esta misma compone la imagen que ya ha sido sometida a un cifrado de cualquier método de criptografía.

Función objetivo

La función objetivo está dada por el Coeficiente de Correlación de Pearson (CCP) que es una medida lineal entre dos variables aleatorias cuantitativas. En palabras comunes, es una forma de medir el grado de relación de dos variables; la ecuación está dada por:

$$\rho_{x,y} = \frac{\sigma_{xy}}{\sigma_x * \sigma_y}$$

Donde:

σ_{xy} es la covarianza de (x, y).

σ_x es la desviación estándar de x.

σ_y es la desviación estándar de y.

La covarianza está dada por:

$$\sigma_{xy} = \frac{1}{n} \sum_{i=1}^{n} x_i y_i - \bar{x} * \bar{y}$$

Donde:

$\bar{x}$ es el valor esperado x o la esperanza de x.

$\bar{y}$ es el valor esperado y o la esperanza de y.

La esperanza está dada por:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

La desviación estándar está dada por:

$$\sigma_r = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}}$$

Para n, en todos los casos, es el número de elementos que tiene x o y.

El Coeficiente de Correlación de Pearson es utilizado como una métrica de evaluación del cifrado para evaluar la calidad del método, este mide la correlación que existe entre los pixeles puesto que en una imagen ordinaria, cada pixel suele estar altamente correlacionado con sus pixeles adyacentes, ya sea dirección horizontal, vertical o diagonal.

Evaluación

Como se acaba de mencionar, la evaluación se hará en tres direcciones tomando el mayor conjunto de pixeles de la misma imagen, es decir, se escoge un conjunto de alelos como una muestra. Con ello se lleva a cabo la valoración que arroja el CCP, siendo ρ_v el valor del CCP en dirección vertical, ρ_h CCP en dirección horizontal y por último ρ_d siendo el CCP en dirección diagonal. Se tomará la suma de los tres como muestra la ecuación:

$$v_i = \rho_v + \rho_h + \rho_d$$

Con ella se hará la comparación entre v_{i-1} y v_i para determinar la calidad del individuo contra la calidad del individuo anterior.

Población

Antes de explicar sobre el tema de población, es importante ver la Genética de Poblaciones donde el matemático Godfrey H. Hardy y del obstetra Wilhelm Weinberg formularon la ley de Hardy-Weinberg en 1908, la cual establece que en una población suficientemente grande, en la que el apareamiento se produce al azar y que no se encuentra sometida a mutación, selección o migración, las frecuencias genéticas y genotípicas permanecen fijas a través de las generaciones y con ello se llega a un equilibrio y la mantiene, quiere decir que la variación genética en una población se mantendrá constante de una generación a la siguiente en ausencia de factores perturbadores.

Es muy importante mantener todos los genes hasta el final ya que, recordemos un principio del cifrado: No haya pérdida de información. Así que, la genética de población de Hardy-Weinberg juega un papel importante. Una vez tomado en cuenta este hecho, el algoritmo se debe adaptar de forma que ningún factor perturbe la información de los alelos para mantenerlos.

Otro factor importante que se observa es que en un Algoritmo Genético, que es usual tener un conjunto de individuos como población, es decir, tratar con un grupo de soluciones para crear una generación nueva de posibles resultados que en prospecto serían mejores. Sin embargo, en este caso, solo se contempla una sola solución, es decir, que el número de individuos en una población es de 1, y ese individuo es la imagen cifrada.

Selección

Por lo mencionado anteriormente se procedio a no tener la selección, para entender el problema que existe en este caso se debe comprender que al implementar algún método de selección se pierden individuos ya que se puede seleccionar más de una vez el mismo individuo quitándole la oportunidad de reproducirse a otros y perdiéndolos en el proceso. La forma de evitar esta pérdida es hacer una permutación de todos los individuos sin que haya ninguno que no esté en el proceso. Se debe aclarar que los individuos en este específico problema son los cromosomas compuestos por los pixeles de la imagen.

Cruza

El método que fue elegido para la reproducción entre individuos fue la cruza mediante el monopunto donde n es la longitud del individuo, en consecuencia $\frac{n}{2}$ es el punto seleccionado para la cruza. De esta forma, se produce una mezcla de pixeles (alelos) entre los cromosomas (individuos) de la imagen de forma que se realiza una recombinación en forma horizontal.

Elitismo

A primera vista pareciera que el método propuesto no tiene elitismo, sin embargo, si lo tiene, de tal forma que hay una selección del mejor entre todas las generaciones que se corren, es decir que para cada generación se guarda la mejor solución procurando comparar al mejor con la solución actual. De ser que la solución actual es mejor, es remplazado el valor que fue mejor hasta ese momento y nombrar el actual el mejor individuo; de ser el caso contrario el valor se mantiene.

Muta

En este caso la muta si se considera pese a lo dicho anteriormente. La muta se manejará de forma que no introduzca nuevos alelos a la población para preservar los genes; en este punto simplemente se recorrerá cada uno de los alelos en la matriz con una cierta probabilidad de muta, en la cual de superar el umbral propuesto se cambiará de posición a otro en el mismo cromosoma, haciendo una recombinación de forma vertical. De este modo, se crea una mutación sin introducir nueva información ni perder la existente.

Remplazo

En esta última etapa, simplemente se integra la nueva descendencia a la población formando la nueva generación, pero como en nuestra población solo tenemos un único individuo se procede a hacer un remplazo total de los genes que han sido modificados por los anteriores si y solo si el actual es mejor que la solución anterior. En este punto sería lo ideal, sin embargo, tenemos el problema de caer en óptimos locales y es por ello que se mete otro operador a la funcionalidad, este ayudará para que esto no pase y que nos permita buscar el óptimo global. Ese operador se pondrá en juego cuando el valor de la solución actual sea peor que la solución pasada, procederá a generar un valor al azar, de ser que este supere el umbral llevará a cabo el remplazo aunque sea una solución peor.

Datos de entrada

La imagen que se utiliza como entrada para el método propuesto es una imagen que haya pasado por cualquier método de cifrado, quiere decir que no es la imagen original sino la imagen cifrada.

5.2. Segunda fase

Para este punto se tiene el mejor resultado, es decir que la imagen tiene otro orden a nivel genético, pero como recordaremos otro punto importante del cifrado es que todos los pasos del proceso sean reversibles. El Algoritmo Genético arroja la nueva imagen y una matriz con la información de la posición de los alelos; por ello se generará un archivo que contendrá esta información de forma que no sea legible sino que sea cifrada. La llave será elegida o generada de forma aleatoria y el archivo será formado con ayuda de la aplicación logística que hemos visto en el capítulo 3; gracias a las propiedades que hemos visto la hace un candidato ideal para el proceso. Para el mismo proceso se involucran otros temas los cuales se verán á continuación.

5.2.1. Disyunción exclusiva

Es un operador lógico también llamado "o exclusivo", se representa como XOR, EOR, EXOR, $\bigoplus$, entre otros. Este operador lógico es usado para una operación en dos valores lógicos. La siguiente tabla, llamada tabla de verdad, muestra los posibles valores de verdad de $p \bigoplus q$, dada cada posible valoración de sus términos de p y q.

p	q	$\bigoplus$
F	F	F
F	T	T
T	F	T
T	T	F

Tabla 5.1: Tabla de verdad del operador XOR
(Tabla propia)

En la tabla 5.1 se dan las posibles combinaciones de dos valores, siendo verdadero (True (T)) y Falso (False (F)), de p y q. Se puede observar que el resultado $(p \bigoplus q)$ muestra falso cuando ambos tienen el mismo valor y solo cuando son diferentes arroja verdadero.

5.2.2. ASCII

ASCII son las siglas para Código Estándar Estadounidense para el Intercambio de Información, es un conjunto de caracteres basado en el alfabeto latino, usando una escala decimal del 0 al 127. Este utiliza 7 bits para representar a los caracteres que se dividen en caracteres imprimibles y de control.

Los caracteres de control no se pensaron originalmente como caracteres imprimibles si no para controlar a los dispositivos como el carácter 10 que representa salto de línea. Los caracteres imprimibles simplemente comprenden a letras, dígitos, signos de puntuación, entre otros símbolos.

La existencia del mismo surge de la necesidad de que las computadoras y dispositivos de diversos fabricantes tuvieran un lenguaje común con el cual comunicarse y ofrecer compatibilidad. Estos números decimales son traducidos por las computadoras en números binarios para ser procesados, por lo tanto, cada una de las letras y algunas órdenes como nueva página corresponden a uno de estos códigos. En la figura 5.2 se muestran los códigos ASCII en decimal $(0 - 255)$, los primeros 128 caracteres son estándar $(0 - 127)$ y los siguientes 128 $(128 - 255)$ son una extensión.

Caracteres de Control ASCII

DEC	HEX	Nombre
0	00	NUL
1	01	SOH
2	02	STX
3	03	ETX
4	04	EOT
5	05	ENQ
6	06	ACK
7	07	BEL
8	08	BS
9	09	HT
10	0A	LF
11	0B	VT
12	0C	FF
13	0D	CR
14	0E	SO
15	0F	SI
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US

Caracteres ASCII Imprimibles

DEC	HEX	Símbolo	DEC	HEX	Símbolo	DEC	HEX	Símbolo
32	20	Espacio	64	40	@	96	60	`
33	21	!	65	41	A	97	61	a
34	22	"	66	42	B	98	62	b
35	23	#	67	43	C	99	63	c
36	24	$	68	44	D	100	64	d
37	25	%	69	45	E	101	65	e
38	26	&	70	46	F	102	66	f
39	27	'	71	47	G	103	67	g
40	28	(	72	48	H	104	68	h
41	29	)	73	49	I	105	69	i
42	2A	*	74	4A	J	106	6A	j
43	2B	+	75	4B	K	107	6B	k
44	2C	,	76	4C	L	108	6C	l
45	2D	-	77	4D	M	109	6D	m
46	2E	.	78	4E	N	110	6E	n
47	2F	/	79	4F	O	111	6F	o
48	30	0	80	50	P	112	70	p
49	31	1	81	51	Q	113	71	q
50	32	2	82	52	R	114	72	r
51	33	3	83	53	S	115	73	s
52	34	4	84	54	T	116	74	t
53	35	5	85	55	U	117	75	u
54	36	6	86	56	V	118	76	v
55	37	7	87	57	W	119	77	w
56	38	8	88	58	X	120	78	x
57	39	9	89	59	Y	121	79	y
58	3A	:	90	5A	Z	122	7A	z
59	3B	;	91	5B	[	123	7B	{
60	3C	<	92	5C	\	124	7C	|
61	3D	=	93	5D	]	125	7D	}
62	3E	>	94	5E	^	126	7E	~
63	3F	?	95	5F	_	127	7F	DEL

ASCII Extendido

DEC	HEX	Símbolo	DEC	HEX	Símbolo	DEC	HEX	Símbolo	DEC	HEX	Símbolo
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	|	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	Pts	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	Espacio

Figura 5.2: Tabla de los caracteres ASCII
(Tecnical)

5.2.3. Formulación del archivo

Para el archivo podemos ver la siguiente imagen (figura 5.3), el cual es un diagrama de flujo que muestra de forma sencilla y general los pasos a seguir para construir el documento que nos ayudará en el proceso de ordenamiento.

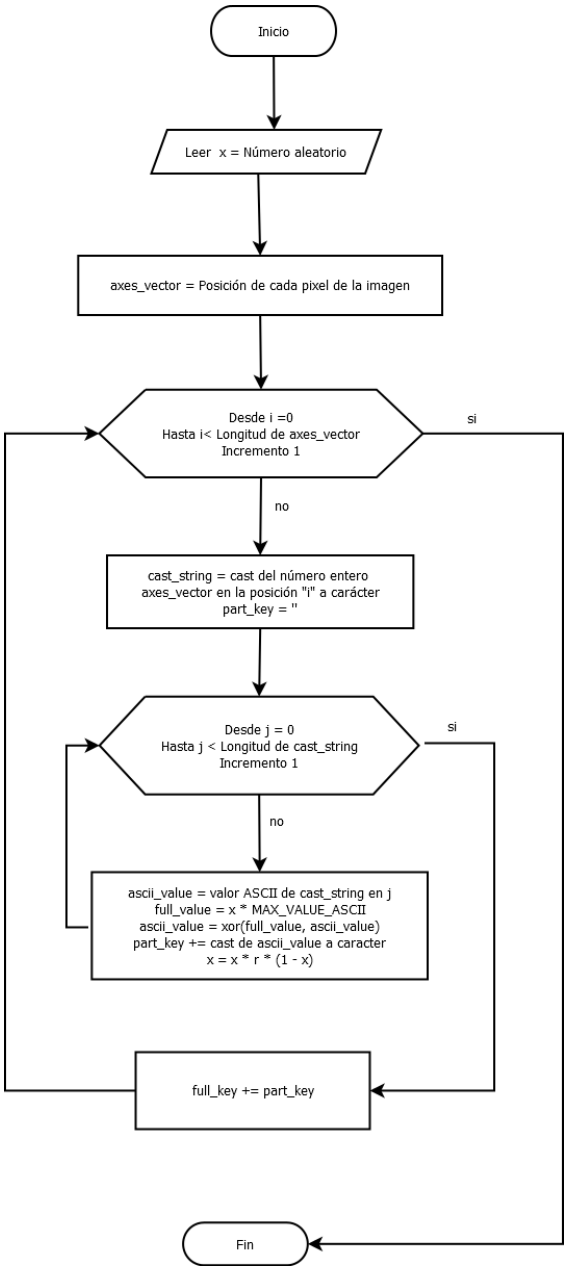


Figura 5.3: Flujo del generador del archivo
(Diagrama propio)

Podemos ver que empieza por leer un número x (llave) el cual será generado de forma aleatoria o se asignará un número específico, eso queda a criterio propio tomando en cuenta que ese número debe ser compartido entre el emisor y el receptor para poder llevar a cabo el proceso de descifrado. Cabe aclarar que la x es la misma que forma parte de la ecuación de la Aplicación Logística (x_n). Después de ello, se toma la información que arroja el algoritmo genético con la posición de los alelos, inmediatamente se empieza con la traducción de cada uno de los caracteres a su correspondiente valor en ASCII.

Seguido se presenta una variable llamada full_value que lleva el valor de la multiplicación entre x y MAX_VALUE_ASCII. MAX_VALUE_ASCII es un número fijo y representa al máximo valor en ASCII que existe o el tope que desea poner para variación de caracteres en el texto. Recordemos que en la tabla de la figura 5.2 existen 255 números decimales, full_values un número dependiente del producto entre x y MAX_VALUE_ASCII que se genera para tomar un número entre 0 a 255 o el tope que se desea elegir para el siguiente paso. Una vez obtenido ese número, se procede a la conversión de decimal a binario para poder ser procesado junto con el valor ascii_value por el operador lógico XOR. Este paso no está en el diagrama de flujo porque el mismo se lleva a cabo en el proceso interno de la función XOR, este operador es aplicado sobre valores binarios tomando como Verdadero a 1 y Falso a 0. La devolución de este resultado se va guardado para generar el texto y proceda a formar un nuevo valor de x que representa a x_{n+1} en la ecuación caótica que entra en juego; esta se utiliza para formar valores aparentemente aleatorios para poder aplicar el operador lógico después de convertirlo en un valor de entre 0 a el valor máximo estipulado, y de esa manera tener distintos valores imposibles de descifrar si no se tiene las condiciones iniciales por las propiedades ya mencionadas anteriormente. De este modo se hace para cada una de los valores en el vector axes_vector.

5.3. Descifrado (Reordenamiento)

Una parte muy importante que siempre es indispensable tener es el descifrado. Este programa simplemente tendrá el proceso de colocar cada pixel que ha sido movido dentro de la misma imagen a su posición original. Esto se lleva a cabo mediante una clave que extraerá de un archivo y con un número inicial que se generó o se estipuló desde el principio entre el receptor y el emisor. Cabe aclarar que este algoritmo es un poco diferente al que genera el archivo (algoritmo que se propone), la diferencia radica en que este no contiene el AG que ayuda a encontrar la mejor solución a condición del número de generaciones que se establezca como parámetro. Este mismo simplemente tiene la meta de llegar al estado inicial (imagen cifrada).

En la figura 5.3 se puede ver el proceso de la generación del archivo. Este proceso también se necesita ya que nos da la información que necesitamos para el reordenamiento de los pixeles, pero se añade una funcionalidad más, se utiliza de la misma manera para poder llegar a los datos originales con los parámetros adecuados. Una vez se hayan obtenido, se procede a ubicar la posición de cada pixel con la información que se arroja, es decir, se reordenan los pixeles en la imagen con la ayuda de este.

Capítulo 6

Resultados

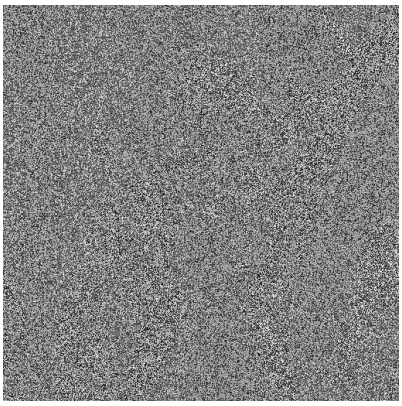
Una parte del algoritmo propuesto que es muy importante y no se ha hecho la debida mención en el capitulo anterior, son los ajustes tanto de operadores como parámetros correspondientes a la primera fase como en la segunda. En este capítulo se mostrarán las diferentes propuestas y resultados que se obtuvieron de cada uno de ellos, manejando seis imágenes diferentes en el proceso. Para el cotejo del algoritmo de cifrado se toma el método del mapa logístico que se ha visto en el capítulo 3.

Las pruebas que se presentan a continuación han sido hechas sobre el sistema operativo Linux Ubuntu-18.04 en una computadora con 8 GB de RAM y un procesador Intel i5-3340 con 2.7 GHz. Así mismo, el algoritmo propuesto fue programado en el lenguaje de alto nivel Python 3.8.

6.1. Lenna

Empezando por la típica imagen de Lenna (Fotografía de Dwight Hooker, 1972) a escala de grises con un tamaño de 512x512, la cual es usada para pruebas de cifrado por sus regiones planas, sombreado, detalles y textura que lo hacen ideal para este tipo de pruebas.

(a) Lenna



(b) Cifrado por aplicación logística

Figura 6.1: Lenna y su correspondiente cifrado

Los valores del cifrado de la imagen de Lenna por el cifrado caótico dejan buenos valores en sí, sobre todo el CCP horizontal. Estos son -0.00185 para la diagonal, $3.62E-05$ para la horizontal y para el vertical 0.000226, el comportamiento del Algoritmo Propuesto que nos interesa, en este caso con esta imagen, ante estas cifras es sobre todo cuando se trata de optimizar la imagen; el resultado del valor horizontal aumenta, esto quiere decir que empeora su valor, pasa por que sacrifica el valor buscando un balance y mejorar a los resultados de la diagonal y la vertical que alcanzan un mejor resultado en conjunto.

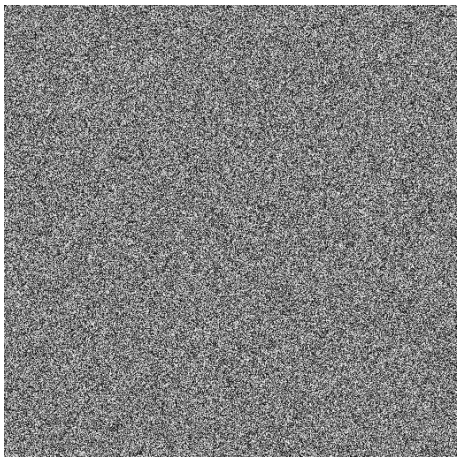
En la tabla 6.1 se muestran los mejores resultados que giran en torno a los parámetros, muestra la efectividad en porcentaje de cada uno de las diferentes evaluaciones que corresponden al coeficiente de correlación entre 2 imágenes, que se pueden ver en las tablas de resultados como la columna imágenes, y entre pixeles de la misma imagen, es decir, las 3 columnas con las direcciones diagonal, horizontal y vertical. También se presenta el tiempo promedio en el que salió el mejor resultado, columna promedio, así como el número de generaciones promedio, de igual manera la generación en la que se encontró el mejor resultado. Se debe aclarar que la columna total es el tiempo que tardo la ejecución de las $1,000$ generaciones y finalmente la columna Gnr, la cual evalúa en total los 4 resultados (diagonal, horizontal, vertical e imágenes).

N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
2	10 %	70 %	20 %	100 %	20 %	100 %	100 %	73 %	80 %	419.2	00:08:17	00:03:18
4	10 %	80 %	10 %	100 %	30 %	70 %	100 %	67 %	75 %	394.1	00:07:52	00:02:55
5	10 %	80 %	20 %	100 %	40 %	80 %	100 %	73 %	80 %	537.1	00:09:07	00:04:39
7	10 %	90 %	10 %	100 %	0 %	90 %	100 %	63 %	73 %	587.4	00:08:06	00:04:31
9	10 %	90 %	30 %	100 %	10 %	90 %	100 %	67 %	75 %	406.5	00:11:32	00:04:30
11	20 %	70 %	20 %	100 %	30 %	70 %	100 %	67 %	75 %	711.2	00:09:47	00:06:36
12	20 %	70 %	30 %	100 %	30 %	70 %	100 %	67 %	75 %	508.5	00:10:14	00:04:58
14	20 %	80 %	20 %	100 %	20 %	70 %	100 %	63 %	73 %	474.5	00:08:55	00:04:00
22	30 %	80 %	10 %	100 %	50 %	90 %	80 %	80 %	80 %	486.2	00:07:12	00:03:16
27	30 %	90 %	30 %	100 %	10 %	60 %	100 %	57 %	68 %	574.2	00:10:59	00:06:02
29	70 %	70 %	20 %	100 %	40 %	90 %	100 %	77 %	83 %	504.3	00:08:31	00:04:03
30	70 %	70 %	30 %	100 %	30 %	80 %	100 %	70 %	78 %	428.4	00:09:55	00:04:02
35	70 %	90 %	20 %	100 %	10 %	80 %	100 %	63 %	73 %	611.4	00:09:06	00:05:17
36	70 %	90 %	30 %	100 %	30 %	90 %	100 %	73 %	80 %	739.2	00:11:26	00:08:13
37	80 %	70 %	10 %	100 %	0 %	80 %	100 %	60 %	70 %	493.4	00:06:58	00:03:12
42	80 %	80 %	30 %	100 %	20 %	90 %	100 %	70 %	78 %	568.7	00:10:57	00:05:57
47	90 %	70 %	20 %	100 %	30 %	60 %	100 %	63 %	73 %	658.2	00:09:01	00:05:37
51	90 %	80 %	30 %	100 %	40 %	100 %	90 %	80 %	83 %	273.3	00:11:16	00:02:57
54	90 %	90 %	30 %	100 %	10 %	80 %	100 %	63 %	73 %	567.1	00:11:15	00:06:07

Tabla 6.1: Mejores resultados de Lenna

Se hicieron 540 pruebas en total, las tablas en este capítulo solo tiene los mejores resultados como se menciono (las tablas completas se pueden ver en el apéndice), 10 ejecuciones se corrieron por cada combinación de operadores. En cada prueba se tomó $1,000$ generaciones con un promedio de tiempo que corresponde a 9 minutos y 0.03 segundos en cada una, es decir, por cada configuración.

En la tabla anterior se puede percibir que los mejores ajustes de parámetros son dos: el número 29 con un Pm de 20 %, Pc de 70 % y Pch 70 % con 504.3 generaciones promedio para un tiempo promedio de 4 minutos y 0.03 segundos para encontrar un buen resultado y el número 51 con un ajuste en parámetros de Pch de 90 %, Pc 80 % Pm 30 % con un promedio de generaciones de 273.3, tiempo promedio de 2 minutos con 57 segundos. En la tabla se presenta que el valor máximo de efectividad en imagen es de 100 %, de pixel 80 % y finalmente en conjunto 83 %, considerando que es muy buena cifra. Se menciono que el resultado que pertenece al CCP horizontal es muy bueno con el cifrado caótico, y esto se puede observar en los resultados, en toda la columna del CCP en dirección diagonal llega fácilmente al 100 %, en CCP dirección vertical también llega al 100 % pero en el CCP en dirección horizontal no puede llegar más allá del 50 %, quiere decir que la mitad de las veces tiene un mejor valor y la otra sacrifica el valor para poder encontrar una solución que beneficie a las otras direcciones, pero esto no es nada critico pues la mayor diferencia que se encontró es de $9.45E-05$ entre el mejor valor y el valor del cifrado caótico.



(a) Cifrado mediante AP

(b) Descifrado de Lenna

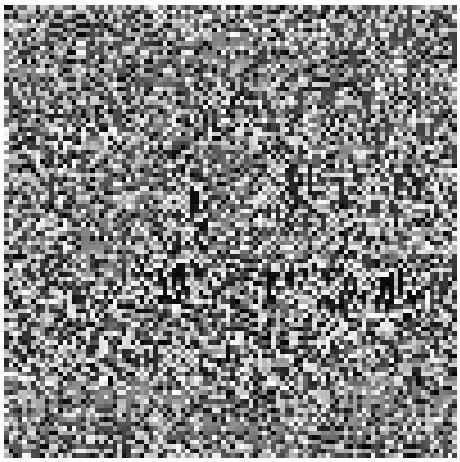
Figura 6.2: Cifrado de Lenna por el AP y su descifrado

En la figura 6.2 se muestra la imagen del correspondiente cifrado por el algoritmo propuesto y la imagen de Lenna al descifrarlo. Ejecutando las pruebas correspondientes al descifrado, cuando se digita la clave erróneamente se presenta un error en la ejecución puesto que la conversión de caracteres en el proceso no se pueden realizar por que la llave es incorrecta, en algunos casos se descifra la imagen completamente pero no llega al estado original, si no, a otro que no permite mostrar el contenido de la imagen, esto es debido a la propiedad de los Sistemas Caóticos. Se tomaron 50 Imágenes cifradas para el descifrado y calcular el tiempo promedio que tarda, el cual es de 27 segundos, resultando en un tiempo menor al cifrado del Algoritmo Propuesto.

6.2. Imagen 100x100

Después de la imagen de Lenna, se tomó la siguiente imagen que mide 100x100 (Imagen del almbum Una Puntata, 2019) para observar el tiempo que tardan las pruebas con respecto a diferentes tamaños. Partiendo del cifrado por el Sistema Caótico (Aplicación logística) se tiene una correlación entre imágenes de 0.0074255, en cuanto a pixeles de la misma imagen en dirección diagonal es de 0.0083966, vertical con 0.001266 y finalmente horizontal con 0.0025229.

(a) 100x100



(b) Cifrado por aplicación logística

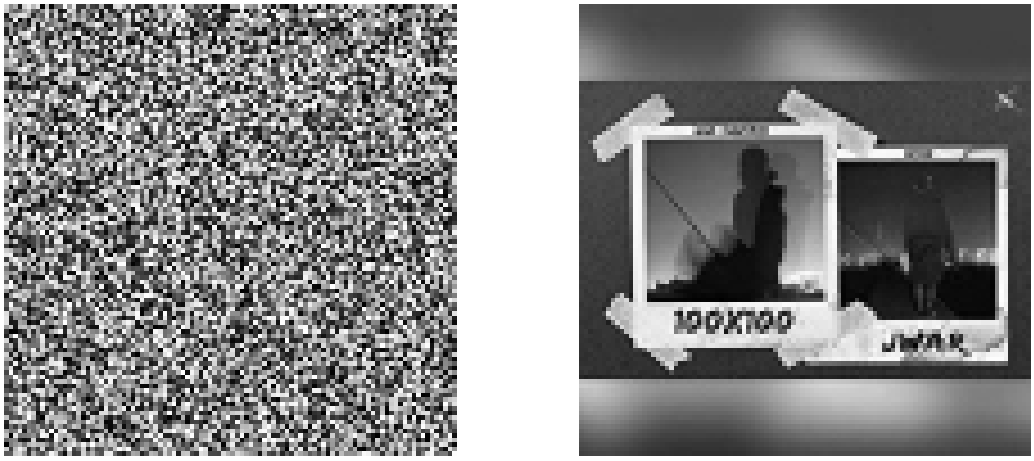
Figura 6.3: 100x100 y su correspondiente cifrado

En la figura 6.3 se muestra una imagen a color, el programa está hecho para tomar ya sea imágenes a color o a escala de grises para poder cifrarlas, puesto que las procesa antes para poder cifrarlas. De esta manera, el resultado es una imagen a escala de grises, debido a que el proceso por ahora maneja solo imágenes de este tipo.

N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
3	10 %	70 %	30 %	100 %	100 %	100 %	60 %	100 %	90 %	449	00:00:25	00:00:11
9	10 %	90 %	30 %	100 %	100 %	100 %	40 %	100 %	85 %	436.4	00:00:26	00:00:11
17	20 %	90 %	20 %	100 %	100 %	100 %	30 %	100 %	83 %	661	00:00:22	00:00:14
21	30 %	70 %	30 %	100 %	100 %	100 %	50 %	100 %	88 %	515	00:00:24	00:00:12
23	30 %	80 %	20 %	100 %	100 %	100 %	60 %	100 %	90 %	457.8	00:00:23	00:00:10
31	70 %	80 %	10 %	100 %	100 %	100 %	60 %	100 %	90 %	532.5	00:00:18	00:00:09
43	80 %	90 %	10 %	100 %	100 %	100 %	70 %	100 %	93 %	512.4	00:00:19	00:00:09
47	90 %	70 %	20 %	100 %	100 %	80 %	80 %	93 %	90 %	528.1	00:00:22	00:00:11
48	90 %	70 %	30 %	100 %	100 %	100 %	30 %	100 %	83 %	358.6	00:00:24	00:00:08

Tabla 6.2: Mejores resultados de la imagen 100x100

Se tomaron 54 configuraciones con 10 pruebas para cada una, sumando un total de 540 corridas con un promedio de 10 segundos. Como se menciono en la tabla 6.2 se presentan los mejores resultados de las 540 pruebas que se hicieron, se puede ver que cada una de los resultados del CCP en cada una de sus direcciones puede alcanzar el porcentaje de 100, sin embargo, le es difícil llegar a ese grado de efectividad con el CCP en dirección vertical aunque eso no es gran problema puesto que el CCP entre los pixeles de una misma imagen perfectamente puede llegar al 100 %, pero se puede ver que el CCP entre imágenes no, con un 80 % de efectividad máximo y esto repercute en el resultado de la columna general llegando solamente al 93 %. El resultado que se encontró con el AP es el número 47 con un Pch de 90 %, Pc 70 % y con un Pm del 30 % este lleva a una efectividad del 100 % en CCP entre los pixeles, un 70 % del CCP entre imágenes y en conjunto un 93 % necesitando un promedio de 513 generaciones tomándose en promedio 0.09 segundos.



(a) Cifrado mediante AP (b) Descifrado de 100x100

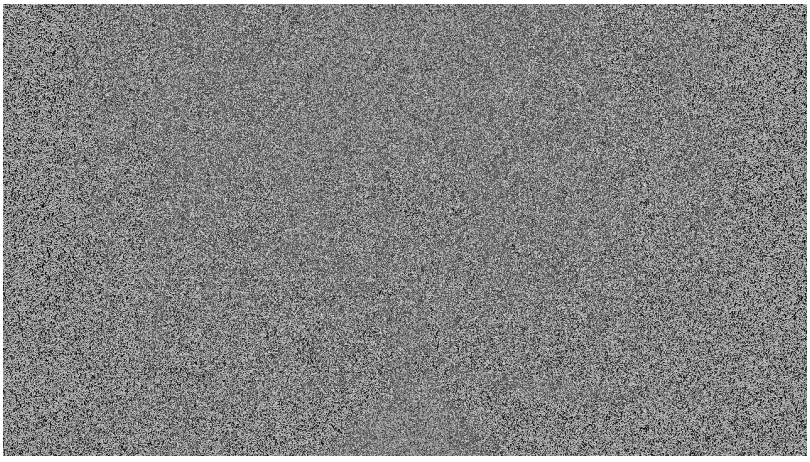
Figura 6.4: Cifrado de 100x100 por el AP y su descifrado

En la figura 6.4 se puede ver el resultado final a escala de grises de la imagen puesto que trabaja de esa manera. El tiempo que toma el código para descifrar esta imagen es de 0.01 segundos, siendo también el tiempo más corto que el cifrado. Se tomaron 50 imágenes cifradas como muestra para el tiempo del descifrado.

6.3. Bosque de bambú de Arashiyama (Kioto, Japón)

La siguiente imagen para analizar es sobre la Imagen del Bosque de bambú en Sagano Arashiyama ubicado en Kyoto (Fotografia de Curt Smith, 2019), con un tamaño de 1280x720, la cual se eligió para diversificar las pruebas probando con paisajes también. Para esta imagen, el coeficiente de correlación del cifrado con respecto a otra imagen es de 0.0014611, con referencia entre pixeles comenzando con dirección diagonal es de -0.000737, horizontal de 0.0011095 y por último vertical de -0.000627.

(a) Bosque de bambú de Sagano



(b) Cifrado por aplicación logística

Figura 6.5: Bosque de bambú de Arashiyama y su correspondiente cifrado

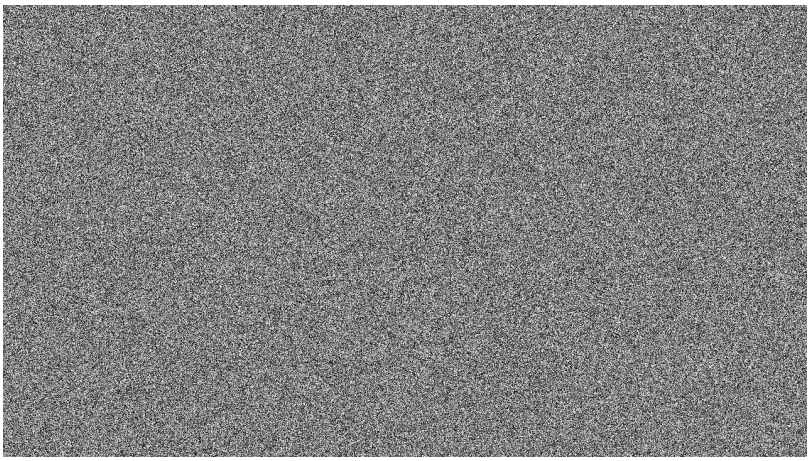
En la imagen cifrada se empiezan a ver vestigios de la imagen anterior siendo necesario un desorden en sus pixeles , por lo que, si un tercero tuviera bajo su poder un conjunto de imágenes que posiblemente el emisor mando o simplemente las consiguió, este podría encontrar la similitud que tiene entre una de ellas. El tema se retomará con ejemplos más visibles como la imagen de la muerte de un miliciano y la imagen de Marilyn Monroe.

N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
3	10 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	448.6	00:34:50	00:14:50
9	10 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	425	00:39:18	00:15:59
22	30 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	554.7	00:27:18	00:14:11
27	30 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	465.1	00:39:11	00:18:00
29	70 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	593.7	00:30:50	00:17:19
35	70 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	459	00:33:18	00:14:31
36	70 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	467.3	00:44:00	00:17:17
44	80 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	376.9	00:35:41	00:12:49
46	90 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	477.3	00:24:54	00:11:03
50	90 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	446.1	00:31:38	00:13:20
51	90 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	480.9	00:38:16	00:17:35
53	90 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	459.3	00:33:25	00:14:33
54	90 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	501.4	00:41:35	00:20:25

Tabla 6.3: Mejores resultados del bosque de bambú

Se ejecutaron 540 pruebas al igual que las imágenes anteriores con 54 configuraciones en sus parámetros y 10 en cada una de las configuraciones con $1,000$ generaciones respectivamente, llevando un tiempo promedio de 32 minutos y 31 segundos por cada corrida. Se puede ver que en cada prueba del CCP se puede llegar al 100 % de efectividad fácilmente y se nota que de 540, 13 pruebas pueden llegar al 100 % de efectividad en todas los tipos de CCP. Con ello en la tabla 6.3 procederemos a tomar el mejor tiempo o el número de generaciones más baja para encontrar al mejor resultado según convenga, para ello podemos ver el numero 46 con un tiempo de 11 minutos y 0.03 segundos tomando como 448 generaciones promedio para encontrar al mejor y una configuración del Pch 90 %, Pc 70 % y un Pm del 20 %. Para el número menor de generaciones se puede ver la prueba número 44 con 377 generaciones promedio y 12 minutos con 49 segundos de tiempo promedio para encontrar al mejor con la configuración de Pch 80 %, con Pc de 90 % y un Pm de 20 %.

Se presenta en la figura 6.6 la imagen descifrada totalmente del paisaje del bosque de bambú que se nota sin pérdida de información y se muestra el cifrado de imagen por el algoritmo que pasa por el AG el cual elimina los vestigios de las imágenes que puedan quedar. Este se tarda en promedio un minuto con 43 segundos, se tomaron de igual forma 50 imágenes para promediar su tiempo.



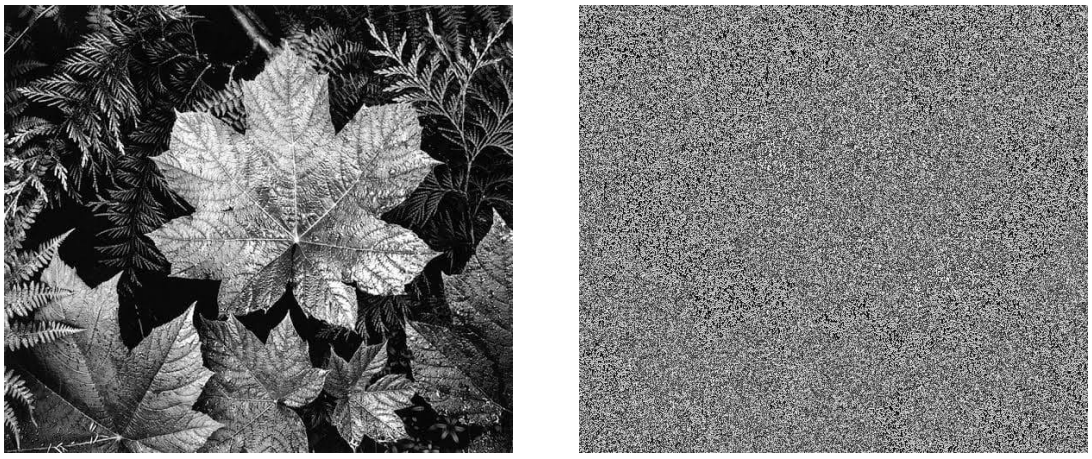
(a) Cifrado mediante AP

(b) Descifrado del bosque de Sagano

Figura 6.6: Cifrado del bosque de Sagano por el AP y su descifrado

6.4. Hojas en el parque nacional Glacier

Fotografía de hojas en primer plano en el parque nacional Glacier (Fotografía de Ansel Easton Adams, 1942), el tamaño de la imagen es de 590x519. Los resultados que se obtuvieron referentes al coeficiente de correlación entre imágenes fueron de 0.0024246 con respecto al CCP entre pixeles de la misma imagen en dirección diagonal es de 0.0003511, horizontal con -0.000208 y en dirección vertical fue de -0.001328.



(a) Hojas en el parque nacional Glacier (b) Cifrado por aplicación logística

Figura 6.7: Hojas en el parque y su correspondiente cifrado

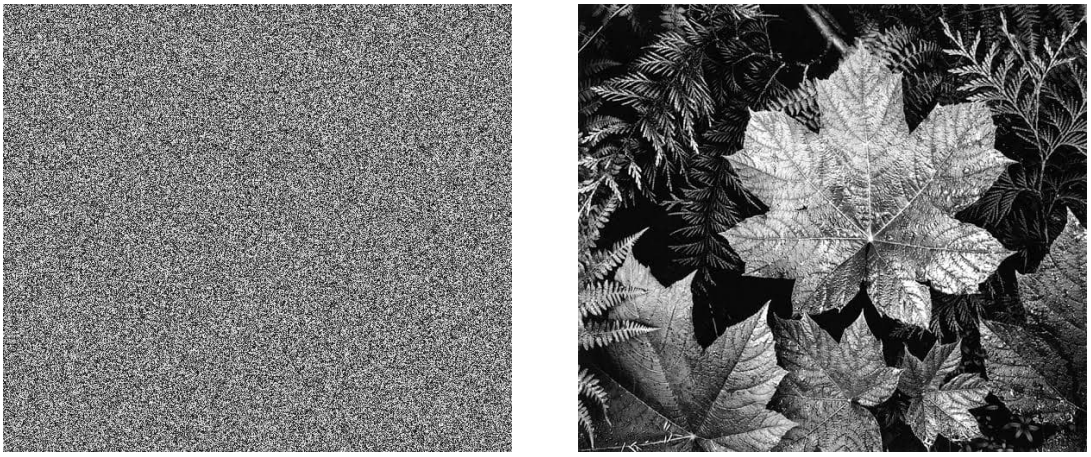
Se hicieron 540 corridas, las cuales se presentan las mejores en la tabla 6.4 con un promedio de 10 minutos y 22 segundos por cada una de ellas con 1000 generaciones. Para esta imagen existe una prueba en la que fácilmente pudo llegar al 100 % de efectividad en todos los casos del CCP, el cual se puede ver como el número 7 con un 10 % de Pch, 90 % de Pc y 10 % Pm, se necesitan 627 generaciones para encontrar una buena solución y un tiempo de 5 minutos y 28 segundos. En las otras soluciones se pueden obtener una buena efectividad pero solo una prueba mostró ser la mejor.

N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
3	10 %	70 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	484.1	00:11:36	00:05:22
4	10 %	80 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	567.2	00:08:35	00:04:34
7	10 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	626.4	00:09:14	00:05:28
18	20 %	90 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	554.8	00:13:21	00:07:06
22	30 %	80 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	520.1	00:08:42	00:04:15
26	30 %	90 %	20 %	90 %	70 %	100 %	100 %	87 %	90 %	572.8	00:11:07	00:06:02

32	70 %	80 %	20 %	100 %	100 %	100 %	70 %	100 %	93 %	508.9	00:10:22	00:04:59
34	70 %	90 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	421.5	00:09:29	00:03:47
37	80 %	70 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	595.5	00:08:10	00:04:33
38	80 %	70 %	20 %	90 %	90 %	100 %	100 %	93 %	95 %	571.3	00:09:32	00:05:08
43	80 %	90 %	10 %	90 %	70 %	100 %	100 %	87 %	90 %	500.7	00:08:49	00:04:09
47	90 %	70 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	631.1	00:10:24	00:06:14
53	90 %	90 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	424.2	00:10:58	00:04:19

Tabla 6.4: Mejores resultados de las hojas en el parque

Se presenta el cifrado por el algoritmo propuesto y su correspondiente cifrado totalmente en la figura 6.8; este se tarda 33 segundos en el descifrado de imagen tomando 50 imágenes cifradas para tomar un promedio del tiempo que tarda este proceso.



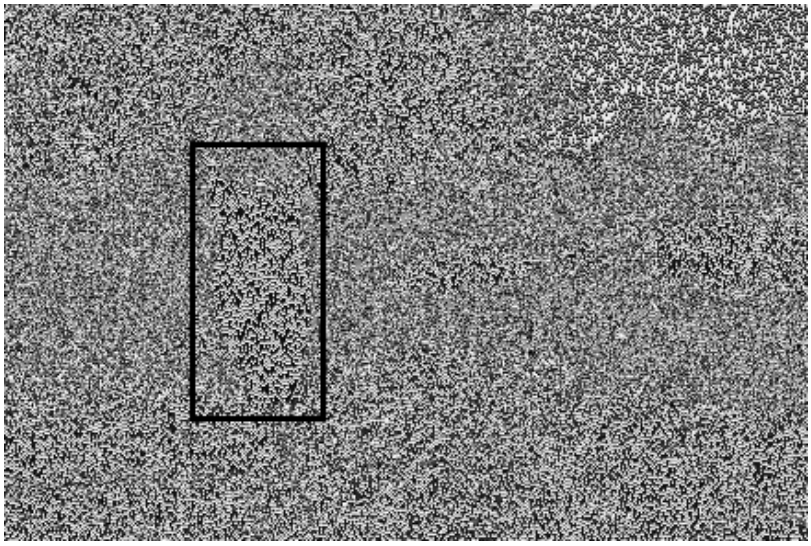
(a) Cifrado mediante AP (b) Descifrado de las hojas del parque

Figura 6.8: Cifrado de las hojas del parque por el AP y su descifrado

6.5. Marilyn Monroe

Fotografía de Marilyn Monroe (Fotografía de Eve Arnold, 1960) con un tamaño de 437x292, con un CCP entre imágenes de 0.0061755. En cuanto al CCP entre pixeles con dirección diagonal tiene -0.004175, horizontal con -0.000266 y con dirección vertical -0.000965.

(a) Marilyn Monroe



(b) Cifrado por aplicación logística

Figura 6.9: Marilyn Monroe y su correspondiente cifrado

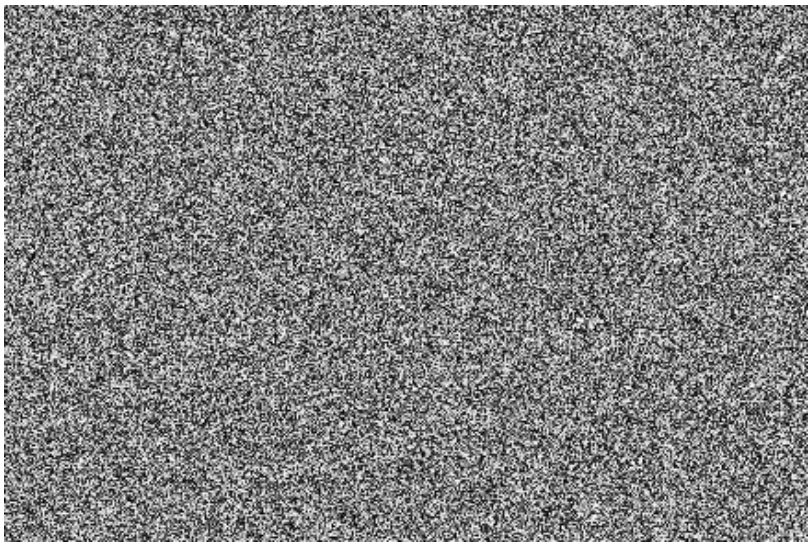
En la figura 6.9b se remarca como el sistema caótico tiene un pequeño inconveniente con algunas partes de la imagen al momento de llevar el proceso de cifrado. En el rectángulo de contorno negro, se hace notar que el vestido de Marilyn se puede ver pese a que la imagen ha sido procesada quedando una visible figura que se puede reconocer como el cuerpo de una mujer.

Se realizaron 540 pruebas de las cuales se presentan los mejores resultados en la tabla 6.5 con un tiempo promedio de 4 minutos y 23 segundos por cada corrida. Por cada configuración se ejecutó 10 corridas para probar el porcentaje de efectividad. Dos resultados obtuvieron 100 % de efectividad en cada prueba dejando la elección en la discriminación por tiempo; el cual corresponde a la prueba número 27 con un Pch de 30 %, Pc de 90 % y un Pm de 30 % contando con 211 generaciones promedio y un tiempo de 1 minuto y 0.04 segundos en encontrar la mejor solución. El segundo resultado que presenta de igual manera una efectividad completa tiene un Pch de 20 %, Pc de 70 % y un Pm de 10 % con un promedio de generaciones de 451 generaciones, siendo más del doble que la prueba 27, y un tiempo de 1 minuto y 23 segundos.

N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
3	10 %	70 %	30 %	100 %	90 %	100 %	100 %	97 %	98 %	601.4	00:04:53	00:02:49
8	10 %	90 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	576.3	00:05:09	00:02:51
10	20 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	450.2	00:03:16	00:01:23
16	20 %	90 %	10 %	100 %	90 %	100 %	100 %	97 %	98 %	582.3	00:03:49	00:02:06
20	30 %	70 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	693.4	00:04:04	00:02:41
26	30 %	90 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	557.1	00:04:48	00:02:33
27	30 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	210.2	00:05:18	00:01:04
40	80 %	80 %	10 %	100 %	90 %	100 %	100 %	97 %	98 %	413.4	00:03:38	00:01:25

Tabla 6.5: Mejores resultados de la imagen de Marilyn

En la figura 6.10a, se muestra que el algoritmo propuesto, se encarga de reordenar los pixeles dado que ningún vestigio de la imagen original se muestre en el cifrado, y en la figura 6.10b se muestra su correspondiente descifrado mostrando que no se ha perdido ni calidad ni información en el proceso. El descifrado de la imagen de Marilyn Monroe se tarda en promedio 13 segundos. De igual manera, se ha tomado una muestra de 50 imágenes para promediar el tiempo que tarda este proceso.



(a) Cifrado mediante AP

(b) Descifrado de Marilyn Monroe

Figura 6.10: Cifrado de Marilyn Monroe por el AP y su descifrado

6.6. Muerte de un miliciano

El punto débil que puede notar al cifrar varias imágenes con el método que propusieron Rasul Enayatifar, Abdul Hanan Abdullah es que al momento de extraer los 5 pixeles donde todos son negros se tiene que:

$$P = [0, 0, 0, 0, 0]$$

Si se hace la conversión a binario del valor de P_1 da: 00000000

Al hacer la conversión de P en ASCII resulta en:

$$B = [0, 0, 0, ..., 0, 0, 0, ..., 0, 0, 0, ..., 0](ASCII)$$

Tal que:

$$U_0 = \frac{0*2^{39}+...+0*2^{28}+...+0*2^{15}+...+0*2^1+0*2^0}{2^{40}} \rightarrow U_0 = 0$$

Posterior a conseguir la llave se procede a cifrar cada uno de los pixeles deforma que se pasa a obtener el nuevo valor del pixel.

$$NuevoValor = round(U_0 * 255) \bigoplus ValorAntiguo$$

Donde el valor antiguo es el valor del pixel actual asignándole 101 $\rightarrow$ 1100101

$$NuevoValor = round(0 * 255) \bigoplus 101 \quad \rightarrow \quad 00000000 \bigoplus 01100101$$

El resultado del NuevoValor es 101, el mismo valor, esto sucede por la tabla de verdad del XOR, cualquier numero n que se le aplique el operador XOR junto con el número 0 resultara en el número n, sin ningún cambio sobre el mismo.

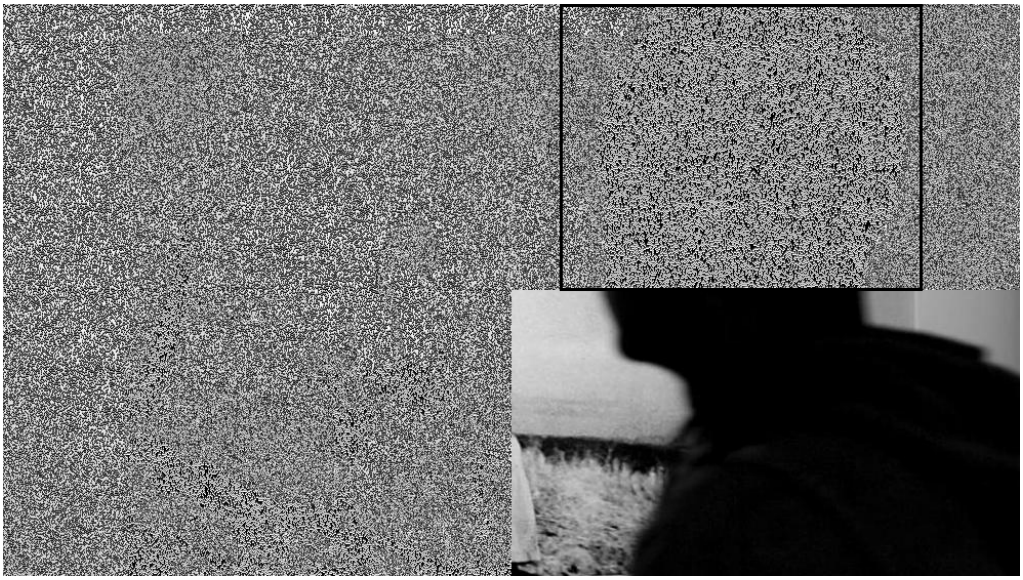
Donde el valor inicial esta dado por $x_0 = U_0$, en la ecuación caótica si remplazamos la llave inicial nos da:

$$x_{(n+1)} = r * 0 * (1 - 0) \quad \rightarrow \quad x_{(n+1)} = 0$$

De esta forma cada uno de los valores posteriores serán 0 y al aplicar el XOR dejara el valor original de cada pixel de tal forma que el cuadrante donde se está llevando a cabo el proceso resultara sin cifrar, en la siguiente figura 6.11b se muestra el ejemplo de este caso.

La siguiente imagen es una fotografía de un visitante frente a la fotografía Roberto Capa de la muerte de un miliciano (Fotografía de Cristobal García, 2016) con un Tamaño 992x558 de la imagen. Esta imagen, al pasar el proceso del cifrado caótico tuvo un CCP entre imágenes de 0.3385045, entre pixeles de la misma imagen cifrada en dirección diagonal de 0.3418875, en horizontal de 0.3405711 y verticalmente de 0.3388259.

(a) Muerte de un miliciano



(b) Cifrado por aplicación logística

Figura 6.11: Muerte de un miliciano y su correspondiente cifrado

Al igual que todas las imágenes se trabajo con 54 configuraciones de las cuales se ejecutaron 10 pruebas en cada configuración formando un total de 540 pruebas para esta imagen, en la cual cada una de estas ejecuciones se tomaron 1000 generaciones para la búsqueda del mejor resultado; se tardo en promedio 31 minutos con 36 segundos para cada una. En la tabla 6.6 se muestra el resumen de sus correspondientes resultados, esta imagen es un caso interesante puesto que, al final del cifrado por la Aplicación Logística no logra cifrar de una manera efectiva y es precisamente por este motivo que al pasar por la imagen cifrada por el Algoritmo Propuesto siempre llega a un cien por ciento de efectividad en cada ejecución ya que el CCP en todos los casos es alto; en el resumen de la tabla se presenta la prueba número 37 con un total de 14 minutos y 38 segundos, el tiempo más bajo que se tuvo entre todas las pruebas con respecto a la ejecución total de las 1000 generaciones y finalmente se presenta la prueba número 5 con un Pch de 10 %, Pc de 80 % y una muta de 20 % con un tiempo promedio de 1 minuto y 21 segundos que necesita para encontrar a la mejor solución y 74 generaciones para ello, esta configuración se considera la mejor; el número de generaciones es el más bajo y el tiempo de igual manera para encontrar al individuo mas apto.

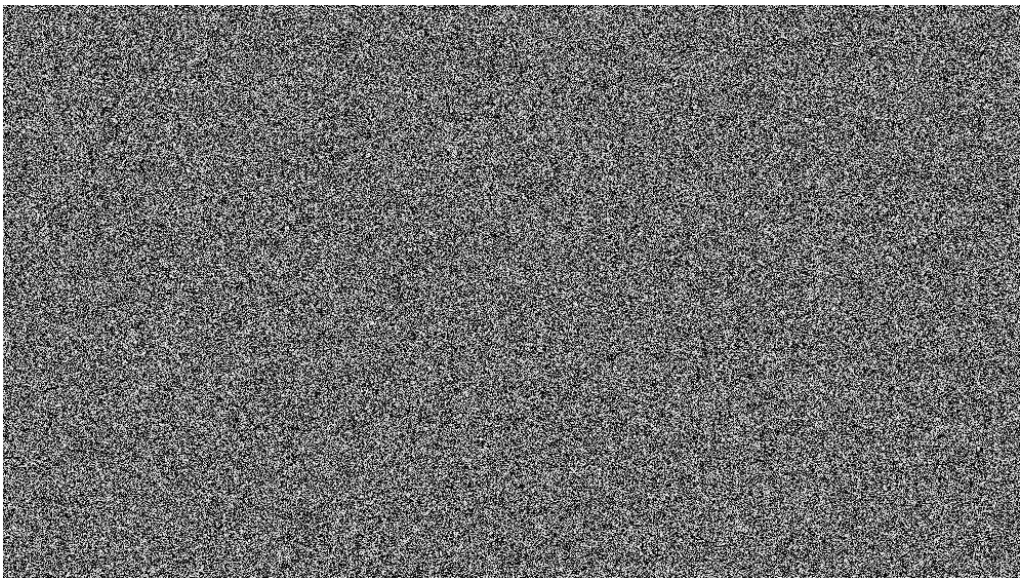
N.	Pch	Pc	Pm	Diagonal	Hrz	Vertical	Imagen	Pixeles	General	Gnr	Total	Promedio
5	10 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	74	00:20:01	00:01:21
37	80 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	644.1	00:14:38	00:08:46

Tabla 6.6: Mejores resultados sobre la imagen de la muerte de un miliciano

En la figura 6.11b ignorando el cuadrante número 4, se hace notar que en el cuadrado con contorno negro, que el cifrado caótico al igual que la imagen de Marilyn Monroe deja un vestigio, en esta se aprecia la forma de una cabeza, mostrando que el cifrado caótico tiene problemas para cifrar debidamente las figuras totalmente negras, pues, cambian su valor, dejan en su lugar una sombra de pixeles en escala de grises que denotan la forma que tenían en la imagen original. En la figura 6.12a se muestra como al pasar por el Algoritmo Propuesto, visiblemente no se encuentra similitud con la original como pasa con el cifrado del sistema caótico con el cuarto cuadrante por el pequeño detalle que se ha explicado anteriormente. Asimismo, tampoco queda ningún vestigio del original en esta figura.

El tiempo promedio que tarda el descifrado del Algoritmo Propuesto es de 1 minuto con 1 segundos. Al igual que las demás, se tomaron 50 imágenes cifradas por el AP para poder tomar un promedio del tiempo. Un caso que se observo al descifrar es que, por la generación de llave del cifrado caótico se contemplo 4 dígitos decimales, considerado muy corto, puesto que al redondearlo de esta manera otro cuadrante aparte del cuarto reflejaba un como llave el cero por este mismo redondeo.

Así que, como solución se amplió el rango a 5 dígitos, el cual se tomaron para cifrar de forma adecuada. Sin embargo, al intentar descifrar la imagen solo se tomaron 4 dígitos para redondear en vez de los 5 que se tomaron para cifrar (fue un error), resultando en un descifrado erróneo, lo cual es una muy buena propiedad del cifrado caótico, pero si es algo en lo que se debe tomar mucha precaución para poder obtener la imagen original.



(a) Cifrado mediante AP

(b) Descifrado de la muerte de un miliciano

Figura 6.12: Cifrado de la muerte de un miliciano por el AP y su descifrado

Capítulo 7

Conclusiones

Es evidente que cada imagen tiene su caso particular y puede tener múltiples soluciones donde puede conseguir el mejor resultado en múltiples configuraciones, siendo el caso de la imagen del bosque de bambú, la imagen de Marilyn Monroe y la imagen de la muerte de un miliciano; o solamente en uno, el cual corresponde a la imagen hojas en primer plano. Hay veces en las que no le es tan fácil llegar al 100 % de efectividad, se contempla para el caso a la imagen de Lenna, hojas en primer plano y la imagen 100x100; o existen casos donde no logra llegar completamente al 100 %, siendo el caso de la imagen 100x100 con un 80 % de efectividad en el CCP entre imágenes y la imagen de Lenna con 80 % de efectividad en cuanto al CCP entre pixeles de la misma imagen repercutiendo en la evaluación general de las dos imágenes. Contemplando esto, se concluye que la mejor configuración con respecto a la correlación entre imágenes, es decir entre la imagen cifrada con el Algoritmo Propuesto y la imagen original, es la ejecución número 47 que se presenta en la siguiente tabla:

N.	PCH	PC	PM	Lenna	100x100	Bosque	Hojas	Marilyn	Miliciano	%	Gnr
47	90 %	70 %	20 %	100 %	80 %	90 %	100 %	100 %	100 %	95 %	548

Tabla 7.1: Resultados del Coeficiente de Correlación entre imágenes

En esta configuración de parámetros, 5 imágenes llegan a su máximo valor entre todas las 540 ejecuciones en cada una excepto la imagen del bosque de bambú, sin embargo, llega a un valor muy aceptable siendo el 90 %, con ello se necesitarían 548 generaciones para encontrar al mejor obteniendo un 95 % en promedio de efectividad.

En cuanto a la correlación entre pixeles de la misma imagen en las direcciones vertical, horizontal y vertical, es la prueba número 22 que se presenta en la siguiente tabla:

N.	PCH	PC	PM	Lenna	100x100	Bosque	Hojas	Marilyn	Miliciano	%	Gnr
22	30 %	80 %	10 %	80 %	97 %	100 %	100 %	90 %	100 %	95 %	484

Tabla 7.2: Resultados del Coeficiente de Correlación entre pixeles

En este caso solo 4 imágenes llegan al máximo valor de entre todas las pruebas excepto la imagen 100x100, pero tiene un muy buen valor de 97 % de efectividad, y Marilyn teniendo un valor aceptable del 90 % de efectividad. Para este caso se necesita en promedio 484 generaciones y la efectividad es de 95 %.

Finalmente la evaluación general que toma en cuenta tanto el CCP entre imágenes y el CCP entre pixeles de la misma imagen que corresponde a la prueba número 3 que se muestra en la siguiente tabla:

N.	PCH	PC	PM	Lenna	100x100	Bosque	Hojas	Marilyn	Miliciano	%	Gnr
3	10 %	70 %	30 %	75 %	90 %	100 %	95 %	98 %	100 %	93 %	524

Tabla 7.3: Resultados sobre la evaluación de ambos

En este caso en particular se tiene un 93 % de efectividad, el más bajo de entre los 3 casos y esto sucede debido a que dos imágenes llegan al máximo valor que tiene entre todas las ejecuciones, se considera que tanto la imagen 100x100 y Marilyn tiene un buen porcentaje de efectividad, en el caso de hojas con 95 % se considera dentro del rango puesto no esta tan alejado al 100 % pero la imagen de Lenna si es el peor, pero esto es debido al CCP que deja el Algoritmo de Enayatifar, Rasul and Abdullah y Abdul Hanan, donde se utilizo la aplicación logística, deja muy buenos valores convirtiéndolo en un cifrado muy bueno en esta imagen. Para este caso se necesitan 524 generaciones en promedio para poder tener el más apto individuo.

Los resultados que se presentan arrojan que el algoritmo se adapta mejor a ciertas circunstancias, es decir cuando el coeficiente de correlación es muy bueno puesto que el cifrado que se uso deja buenos valores entonces es mejor no usarlo a menos que visualmente deje vestigios en la imagen cifrada, dicho en otras palabras el Algoritmo Propuesto es mejor aprovechado cuando el coeficiente de correlación no tiene un buen valor (es alto es decir tendiendo a 1) o/y la imagen cifrada deja a relucir figuras que denotan partes de la imagen original.

Las pruebas que se realizaron referentes al parámetro de cambio, mencionado en el capítulo 5, contemplaron principalmente 70 %, 80 % y 90 % en probabilidad para que no se cambie el mejor individuo por el peor, sin embargo, se hicieron pruebas con 10 %, 20 % y 30 % para tener más probabilidad de buscar entre otros individuos y no caer en óptimos locales y los resultados en cuanto el CCP global y el CCP entre pixeles demostraron que: Es mejor buscar entre generaciones el mejor individuo y mejorar en el proceso el actual elegido que quedarse con el mejor la mayor parte del tiempo y tratar de mejorarlo para que sea el más apto.

Se puede ver que el tamaño de la imagen influye en gran medida en el tiempo, entre mayor tamaño tenga la imagen mayor será el tiempo, esto se ve en la siguiente tabla:

Tamaño	100x100	437x292	512x512	590x519	992x558	1280x720
Imagen	100x100	Marilyn	Lenna	Hojas	Miliciano	Bosque
Máximo	00:00:15	00:03:49	00:08:13	00:07:47	00:15:25	00:20:48
Mínimo	00:00:06	00:01:04	00:02:22	00:02:50	00:01:21	00:05:48

Tabla 7.4: Tiempo de cifrado

En la tabla 7.4 se demuestra el tiempo máximo y mínimo que tardo la imagen en encontrar al individuo más apto y el máximo es de 20 minutos esto es demasiado tiempo y es una desventaja muy grande aun cuando se contemple al tiempo mínimo de 5 minutos para una imagen de 1280x720 es demasiado.

Al descifrar, el tiempo que tarda al realizar la tarea es definitivamente más corto, el cual se puede ver en la tabla 7.5, el tiempo promedio para descifrar la imagen más grande (1280x720) es de 1 minuto 43 segundos, es bastante bajo a comparación al cifrado; pero aun así es un tiempo considerable.

Tamaño	100x100	437x292	512x512	590x519	992x558	1280x720
Imagen	100x100	Marilyn	Lenna	Hojas	Miliciano	Bosque
Tiempo	00:00:01	00:00:13	00:00:27	00:00:33	00:01:01	00:01:43

Tabla 7.5: Tiempo de descifrado

Considero que el método propuesto es bueno bajo circunstancias específicas, expuestas en el presente trabajo; un ejemplo es cuando el recurso del tiempo de cifrado no sea tan importante, se considera que el método es fácil de implementar en cualquier entorno y lenguaje de programación así como acoplar a algún cifrado o tenerlo en un archivo por aparte para aplicarlo en las situaciones correspondientes.

7.1. Propuesta futura

Para trabajo posterior se pueden tomar algunas consideraciones futuras para mejorar algunos puntos que son de vital relevancia para el trabajo, un punto que se plantea al ver los tiempos que tarda el algoritmo que se propone, es el equilibrio que se puede tomar entre costo computacional vs calidad, esto es sobre tomar un conjunto de pixeles para el algoritmo en vez de tomarlos de forma individual para reducir el tiempo que toma. La forma de rastrear la posición de los pixeles no es el mejor método empleado por que se podría calcular la posición de cada uno de ellos y sustituir el archivo por una ecuación simple para deducirlo y no pasarlos por un archivo.

Como es bien sabido para este punto el trabajo resulta en un anagrama de pixeles, es decir, se realiza una trasposición de pixeles en la misma imagen, se podría proponer como un método de cifrado completo por si mismo añadiendo el cambio del valor de cada pixel, con ello se deberían hacer sus correspondientes métricas para saber la fuerza que tiene al cifrar y ver si esa propuesta es viable.

Se podrían implementar hilos en el proyecto de manera futura como un extra para posiblemente poder trabajar con el método en tiempo real, en las condiciones actuales no es posible por el tiempo promedio que tarda optimizando.

Apéndice A

Código Fuente

A.1. Sistema de cifrado\descifrado mediante aplicación logistica

```
from PIL import Image


ORIGINAL_IMAGE = "Original_Image.jpg" #"Decrypted.png"
ENCRYPTED_IMAGE = "Encrypted.png" #"Decrypted_Ag.png"
QUADRANTS = 4
KEY_PIXEL_NUMBER = 5
BASE = 2
BINARY_LENGTH = 8
ROUND = 4
TOTAL_BINARY_LENGTH = BINARY_LENGTH * KEY_PIXEL_NUMBER


def xor(key, pixels):
    binary_value = ""
    cont = 0
    if len(list(str(int(bin(key)[2:])))) >
        len(list(str(int(bin(pixels)[2:])))):
        # binary pixel
        longer_binary = list(str(int(bin(key)[2:])))
        # binary key
        smaller_binary = list(str(int(bin(pixels)[2:])))
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    else:
        # binary pixel
        longer_binary = list(str(int(bin(pixels)[2:])))
        # binary key
        smaller_binary = list(str(int(bin(key)[2:])))
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    for i in range(0, larger_size):
        if larger_size - smaller_size <= i:
            binary_value += str(int(bool(int(longer_binary[i]))
```

```
                != bool(int(smaller_binary[cont])))
            cont = cont + 1
        else:
            binary_value += str(int(bool(int(longer_binary[i]))
                != bool(0)))
    pixel_value = int(binary_value, 2)
    return pixel_value


def key_generator(final, pixels):
    pixel_value = []
    row = final[1]
    column = final[0]
    generated_key = 0
    for i in range(0, KEY_PIXEL_NUMBER):
        pixel_value.append(int(pixels[column, row]))
        column = column-1
        row = row - 1
    for i in range(0, len(pixel_value)):
        binary_pixel = list(str(int(bin(pixel_value[i])[2:])))
        # valor del pixel 255 (8 bits) starts from 2^0 ends 2^39
        # count 0 to 39
        cont = i * BINARY_LENGTH
        for j in range(len(binary_pixel) - 1, -1, -1):
            # summation (pixel binary * 2 ^ n[ 0, 39 ])
            generated_key = generated_key + (pow(BASE, cont)
                * int(binary_pixel[j]))
            cont = cont + 1

    # summation / 2 ^ 40
    generated_key = round(generated_key / float(pow(BASE,
        TOTAL_BINARY_LENGTH)), ROUND)
    return  generated_key


def operation(start, final, pixels, key):
    for x in range (int(start[0]),int(final[0]) + 1):
        for y in range (int(start[1]), int(final[1]) + 1):
            if (x == final[0] and y == final[1]) or
              (x == (final[0] - 1) and
              y == (final[1] - 1)) or (x == (final[0] - 2) and
              y == (final[1] - 2)) or (x == (final[0] - 3) and
              y == (final[1] - 3)) or (x == (final[0] - 4) and
              y == (final[1] - 4)) :
                continue
            full_key = int(key * 255)
            pixels[x, y]= xor(full_key, pixels[x, y])
            # chaotic function
            key = key * QUADRANTS * (1 - key)
    return pixels


def main():
    try:
        image = Image.open(ORIGINAL_IMAGE).convert("L")
```

```
        column, row = image.size
        pixels = image.load()
        hxw = {0: [[0, 0], [column / 2 - 1, row / 2 - 1]],
               1:[[column / 2, 0], [column - 1, row / 2 - 1]],
               2:[[0, row / 2], [column / 2 - 1, row - 1]],
               3:[[(column / 2), (row / 2)],
               [column - 1, row - 1]]}
        for i in range(0, QUADRANTS):
            start, final = hxw[i]
            key = key_generator(final, pixels)
            print ("Key number ",(i + 1),": ", key)
            pixels = operation(start, final, pixels, key)
        image.save(ENCRYPTED_IMAGE)
        image.close()
        print ("\n\nIt's done!!!.....\n\n")
    except ValueError:
        print ("It was not possible to convert")


main()
```

A.2. Algoritmo Propuesto

```
import imageio
from random import sample
import numpy as np
import random
import copy

ENCRYPTED_AG_IMAGE = "Encrypted_Ag.png"
ENCRYPTED_IMAGE = "Encrypted.png"
ORIGINAL_IMAGE = "Original_Image_BN.png"
KEY_TEXT = 'key.txt'
PC = 0.7
PM = 0.2
#   Probability of change
PCH = 0.9
ITERATIONS = 0
R = 4
ROUND = 4
# maximum ASCII value
MAX_VALUE_ASCII = 255

cipher_correlation_diagonal = 0.0
cipher_correlation_horizontal = 0.0
cipher_correlation_vertical = 0.0
best_correlation_diagonal = 1000000000000.0
best_correlation_horizontal = 1000000000000.0
best_correlation_vertical = 1000000000000.0

best_solution = []
select_matrix = []
```

```
current_axes_pixels = []
best_axes_pixels = []
previous_axes_pixels = []

def matrix_to_horizontal_vectors(image):
    vector = []
    matrix = []
    row = len(image)
    column = len(image[0])
    vector = image[0:row, 0:(column - 1)]
    matrix.append(vector.flatten())
    vector = image[0:row, 1:column]
    matrix.append(vector.flatten())
    return np.corrcoef(matrix[0], matrix[1])[0, 1]


def matrix_to_vertical_vectors(image):
    vector = []
    matrix = []
    row = len(image)
    column = len(image[0])
    vector = image[0:(row - 1), 0:column]
    matrix.append(vector.flatten())
    vector = image[1:row, 0:column]
    matrix.append(vector.flatten())
    return np.corrcoef(matrix[0], matrix[1])[0, 1]


def matrix_to_diagonal_vectors(image):
    vector = []
    matrix = []
    row = len(image)
    column = len(image[0])
    vector = image[0:(row - 1), 0:(column - 1)]
    matrix.append(vector.flatten())
    vector = image[1:row, 1:column]
    matrix.append(vector.flatten())
    return np.corrcoef(matrix[0], matrix[1])[0, 1]


def variable_initialization(cipher, row, column):
    global cipher_correlation_diagonal
    global cipher_correlation_horizontal
    global cipher_correlation_vertical
    global current_axes_pixels
    global best_axes_pixels
    global previous_axes_pixels
    cipher_correlation_diagonal =
                        matrix_to_diagonal_vectors(cipher)
    cipher_correlation_horizontal =
                        matrix_to_horizontal_vectors(cipher)
    cipher_correlation_vertical =
                        matrix_to_vertical_vectors(cipher)
    current_axes_pixels = np.zeros((row,column),dtype=int)
    best_axes_pixels = np.zeros((row,column),dtype=int)
```

```
    previous_axes_pixels = np.zeros((row,column),dtype=int)
    x = 0
    y = 0
    for i in range((row*column)):
        current_axes_pixels[x][y] = i
        best_axes_pixels[x][y] = i
        previous_axes_pixels[x][y] = i
        if (y+1) < column:
            y += 1
        else:
            x += 1
            y = 0



def mutation(current_pixels, rows, column):
    global current_axes_pixels
    for i in range(0, rows):
        for j in range(0, column):
            if random.uniform(0, 1) <= PM:
                while True:
                    mutation_number =
                            random.randint(0, (column - 1))
                    if mutation_number != j:
                        break
                temporary_value = current_pixels[i][j]
                current_pixels[i][j] =
                        current_pixels[i][mutation_number]
                current_pixels[i][mutation_number] =
                                            temporary_value
                temporary_value = current_axes_pixels[i][j]
                current_axes_pixels[i][j] =
                    current_axes_pixels[i][mutation_number]
                current_axes_pixels[i][mutation_number] =
                                                temporary_value
    return current_pixels


def crossover(current_pixels, column):
    global current_axes_pixels
    temporary_pixels = copy.deepcopy(current_pixels)
    temporary_axes_matrix = copy.deepcopy(current_axes_pixels)
    len_matrix = (len(select_matrix)/2)*2
    for i in range(0, len(select_matrix), 2):
        if (random.uniform(0, 1) <= PC) and (i < len_matrix):
            for j in range(0, column):
                if j < (column / 2):
                    # original array
                    temporary_pixels[i][j] =
                       current_pixels[select_matrix[i]][j]
                    # axes (x,y
                    temporary_axes_matrix[i][j] =
                    current_axes_pixels[select_matrix[i]][j]
                    # original array
                    temporary_pixels[i + 1][j] =
```

```
                    current_pixels[select_matrix[i + 1]][j]
                # axes (x,y)
                temporary_axes_matrix[i + 1][j] =
                current_axes_pixels[select_matrix[i + 1]][j]
            else:
                # original array
                temporary_pixels[i + 1][j] =
                     current_pixels[select_matrix[i]][j]
                #axes (x,y)
                temporary_axes_matrix[i + 1][j] =
                current_axes_pixels[select_matrix[i]][j]
                # original array
                temporary_pixels[i][j] =
                 current_pixels[select_matrix[i + 1]][j]
                # axes (x,y)
                temporary_axes_matrix[i][j] =
                current_axes_pixels[select_matrix[i + 1]][j]
        else:
            # original array
            temporary_pixels[i] =
               copy.deepcopy(current_pixels[select_matrix[i]])
            # axes (x,y)
            temporary_axes_matrix[i] =
            copy.deepcopy(current_axes_pixels[select_matrix[i]])
            if (i + 1) < len(select_matrix):
                # original array
                temporary_pixels[i + 1] =
                copy.deepcopy(current_pixels[select_matrix[i + 1]])
                # axes (x,y)
                temporary_axes_matrix[i + 1] =
                copy.deepcopy(current_axes_pixels[select_matrix[i + 1]])
    current_axes_pixels = copy.deepcopy(temporary_axes_matrix)
    return temporary_pixels


#   selection permutation
def selection(size_row):
    global select_matrix
    select_matrix = sample([x for x in range(0, size_row)], size_row)



def objective_function(current_pixels, previous_pixels):
    global best_correlation_diagonal
    global best_correlation_horizontal
    global best_correlation_vertical
    global best_solution
    global current_axes_pixels
    global best_axes_pixels
    global previous_axes_pixels
    current_correlation_diagonal =
              matrix_to_diagonal_vectors(current_pixels)
    current_correlation_horizontal =
             matrix_to_horizontal_vectors(current_pixels)
    current_correlation_vertical =
```

```
            matrix_to_vertical_vectors(current_pixels)
    previous_correlation_diagonal =
            matrix_to_diagonal_vectors(previous_pixels)
    previous_correlation_horizontal =
            matrix_to_horizontal_vectors(previous_pixels)
    previous_correlation_vertical =
            matrix_to_vertical_vectors(previous_pixels)
    current_correlation = abs(current_correlation_diagonal) +
    abs(current_correlation_horizontal) +
                          abs(current_correlation_vertical)
    previous_correlation = abs(previous_correlation_diagonal) +
    abs(previous_correlation_horizontal) +
                          abs(previous_correlation_vertical)
    best_correlation = abs(best_correlation_diagonal) +
    abs(best_correlation_horizontal) +
                           abs(best_correlation_vertical)
    if previous_correlation < current_correlation:
        if best_correlation > previous_correlation:
            best_solution = copy.deepcopy(previous_pixels)
            best_correlation_diagonal =
                           previous_correlation_diagonal
            best_correlation_horizontal =
                          previous_correlation_horizontal
            best_correlation_vertical =
                            previous_correlation_vertical
            best_axes_pixels = copy.deepcopy(previous_axes_pixels)
        if random.uniform(0, 1) < PCH:
            current_pixels = copy.deepcopy(previous_pixels)
            current_axes_pixels = copy.deepcopy(previous_axes_pixels)
    else:
        if best_correlation > current_correlation:
            best_solution = copy.deepcopy(current_pixels)
            best_correlation_diagonal =
                            current_correlation_diagonal
            best_correlation_horizontal =
                          current_correlation_horizontal
            best_correlation_vertical =
                            current_correlation_vertical
            best_axes_pixels = copy.deepcopy(current_axes_pixels)
    previous_axes_pixels = copy.deepcopy(current_axes_pixels)
    return current_pixels


def clean():
    global select_matrix
    select_matrix = []


def xor(key, pixels):
    binary_value = ""
    cont = 0
    if len(list(str(int(bin(key)[2:])))) >
                    len(list(str(int(bin(pixels)[2:])))):
        # binary pixel
        longer_binary = list(str(int(bin(key)[2:])))
```

```
        # binary key
        smaller_binary = list(str(int(bin(pixels)[2:])))
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    else:
        # binary pixel
        longer_binary = list(str(int(bin(pixels)[2:])))
        # binary key
        smaller_binary = list(str(int(bin(key)[2:])))
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    for i in range(0, larger_size):
        if larger_size - smaller_size <= i:
            binary_value += str(int(bool(int(longer_binary[i]))
                                    != bool(int(smaller_binary[cont]))))
            cont = cont + 1
        else:
            binary_value += str(int(bool(int(longer_binary[i]))
                                                != bool(0)))
    pixel_value = int(binary_value, 2)
    return pixel_value


def key_generator(x):
    full_key = ""
    axes_vector = best_axes_pixels.flatten()
    for i in range(len(axes_vector)):
        cast_string = str(axes_vector[i])
        if (i + 1) < len(axes_vector):
            cast_string += " "
        part_key = ""
        for j in cast_string:
            ascii_value = ord(j)
            full_value = int(x * MAX_VALUE_ASCII)
            ascii_value = xor(full_value, ascii_value)
            character = chr(ascii_value)
            part_key += character
            # chaotic function
            x = x * R * (1 - x)
        full_key += part_key
    return full_key


def genetic_algorithm():
    global ITERATIONS
    ITERATIONS = int(input("Generaciones: "))
    current_pixels = np.array(imageio.imread(ENCRYPTED_IMAGE))
    previous_pixels = copy.deepcopy(current_pixels)
    size_row = len(current_pixels)
    size_column = len(current_pixels[0])
    variable_initialization(current_pixels, size_row, size_column)
    for i in range(0, ITERATIONS):
        current_pixels =
        objective_function(current_pixels, previous_pixels)
        previous_pixels = copy.deepcopy(current_pixels)
```

```
        selection(size_row)
        current_pixels = crossover(current_pixels, size_column)
        current_pixels =
            mutation(current_pixels, size_row, size_column)
        clean()
    # (Name, Matrix)
    imageio.imwrite(ENCRYPTED_AG_IMAGE, best_solution)


def key_file_generator():
    original_key = round(random.random(), ROUND)
    print("clave: ",original_key)
    key = original_key
    full_text = key_generator(key)
    #key_file = open(KEY_TEXT, 'wb')
    #key_file.write(full_text)
    with open(KEY_TEXT, 'wb') as f:
        f.write(full_text.encode())
    #key_file.close()


def main():
        genetic_algorithm()
        key_file_generator()
        print ("\n\nIt's done!!!.....\n\n")


main()
```

A.3. Descifrado del Algoritmo Propuesto

```
import imageio
import numpy as np
import copy

KEY_FILE = "key.txt"
R = 4
ROUND = 4
MAX_VALUE_ASCII = 255
ENCRYPTED_AG_IMAGE = "Encrypted_Ag.png"
DECRYPTED_AG_IMAGE = "Decrypted_Ag.png"


def xor(key, pixels):
    binary_value = ""
    cont = 0
    if len(list(str(int(bin(key)[2:])))) >
                        len(list(str(int(bin(pixels)[2:])))):
        # binary pixel
        longer_binary = list(str(int(bin(key)[2:])))
        # binary key
        smaller_binary = list(str(int(bin(pixels)[2:])))
```

```
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    else:
        # binary pixel
        longer_binary = list(str(int(bin(pixels)[2:])))
        # binary key
        smaller_binary = list(str(int(bin(key)[2:])))
        larger_size = len(longer_binary)
        smaller_size = len(smaller_binary)
    for i in range(0, larger_size):
        if larger_size - smaller_size <= i:
            binary_value += str(int(bool(int(longer_binary[i])) !=
                                    bool(int(smaller_binary[cont]))))
            cont = cont + 1
        else:
            binary_value += str(int(bool(int(longer_binary[i])) !=
                                                        bool(0)))
    pixel_value = int(binary_value, 2)
    return pixel_value


def pixel_generator(key):
    full_text = ""
    #file_key = open(KEY_FILE, "rb")
    file_key = open(KEY_FILE, "rb")
    text = file_key.read().decode(encoding="utf-8")
    for i in text:
        #print(i)
        #input("some")
        ascii_value = ord(i)
        full_key = int(key * MAX_VALUE_ASCII)
        ascii_value = xor(full_key, ascii_value)
        full_text += chr(ascii_value)
        # chaotic function
        key = key * R * (1 - key)
    file_key.close()
    return full_text


def string_to_array_integers(string):
    list = string.split(" ")
    vector = [int(i) for i in list]
    return vector


def generated_image(pixels):
    current_pixels = np.array(imageio.imread(ENCRYPTED_AG_IMAGE))
    size_row = len(current_pixels)
    size_column = len(current_pixels[0])
    temporary_current_pixels = current_pixels.flatten()
    order_vector = []
    for i in range(len(temporary_current_pixels)):
        order_vector.append((pixels[i], temporary_current_pixels[i]))
    order_vector.sort()
    k = -1
```

```
    for i in range(size_row):
        column = []
        for j in range(size_column):
            k += 1
            column.append(order_vector[k][1])
        current_pixels[i] = copy.deepcopy(column)
    imageio.imwrite(DECRYPTED_AG_IMAGE, current_pixels)


def main():
    key = float(input("Key: "))
    pixel_generated = pixel_generator(key)
    pixel_generated = string_to_array_integers(pixel_generated)
    generated_image(pixel_generated)
    print ("\n\nIt's done!!!.....\n\n")


main()
```

Apéndice B

Tabla de Resultados

B.1. Lenna

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	100 %	20 %	90 %	90 %	70 %	75 %	647.1	00:06:59	00:04:14
2	10 %	70 %	20 %	100 %	20 %	100 %	100 %	73 %	80 %	419.2	00:08:17	00:03:18
3	10 %	70 %	30 %	100 %	20 %	90 %	90 %	70 %	75 %	550.8	00:09:51	00:05:10
4	10 %	80 %	10 %	100 %	30 %	70 %	100 %	67 %	75 %	394.1	00:07:52	00:02:55
5	10 %	80 %	20 %	100 %	40 %	80 %	100 %	73 %	80 %	537.1	00:09:07	00:04:39
6	10 %	80 %	30 %	100 %	0 %	80 %	70 %	60 %	63 %	534.2	00:11:21	00:05:48
7	10 %	90 %	10 %	100 %	0 %	90 %	100 %	63 %	73 %	587.4	00:08:06	00:04:31
8	10 %	90 %	20 %	100 %	20 %	90 %	80 %	70 %	73 %	468.8	00:09:37	00:04:17
9	10 %	90 %	30 %	100 %	10 %	90 %	100 %	67 %	75 %	406.5	00:11:32	00:04:30
10	20 %	70 %	10 %	100 %	20 %	90 %	90 %	70 %	75 %	338.9	00:07:28	00:02:22
11	20 %	70 %	20 %	100 %	30 %	70 %	100 %	67 %	75 %	711.2	00:09:47	00:06:36
12	20 %	70 %	30 %	100 %	30 %	70 %	100 %	67 %	75 %	508.5	00:10:14	00:04:58
13	20 %	80 %	10 %	100 %	20 %	100 %	90 %	73 %	78 %	681.9	00:07:31	00:04:48
14	20 %	80 %	20 %	100 %	20 %	70 %	100 %	63 %	73 %	474.5	00:08:55	00:04:00
15	20 %	80 %	30 %	100 %	20 %	70 %	70 %	63 %	65 %	559.8	00:10:39	00:05:42

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
16	20 %	90 %	10 %	100 %	30 %	90 %	90 %	73 %	78 %	461.8	00:07:38	00:03:19
17	20 %	90 %	20 %	100 %	10 %	70 %	80 %	60 %	65 %	340.8	00:09:32	00:03:06
18	20 %	90 %	30 %	100 %	10 %	60 %	70 %	57 %	60 %	449.3	00:11:45	00:05:04
19	30 %	70 %	10 %	100 %	30 %	60 %	80 %	63 %	68 %	627.3	00:07:03	00:04:09
20	30 %	70 %	20 %	100 %	30 %	90 %	90 %	73 %	78 %	565.3	00:08:40	00:04:38
21	30 %	70 %	30 %	100 %	50 %	80 %	90 %	77 %	80 %	554.8	00:10:28	00:05:32
22	30 %	80 %	10 %	100 %	50 %	90 %	80 %	80 %	80 %	486.2	00:07:12	00:03:16
23	30 %	80 %	20 %	100 %	10 %	70 %	80 %	60 %	65 %	535.1	00:08:57	00:04:33
24	30 %	80 %	30 %	100 %	40 %	70 %	90 %	70 %	75 %	519.3	00:10:38	00:05:17
25	30 %	90 %	10 %	100 %	20 %	100 %	70 %	73 %	73 %	625.9	00:07:53	00:04:37
26	30 %	90 %	20 %	100 %	20 %	60 %	90 %	60 %	68 %	489.7	00:09:40	00:04:30
27	30 %	90 %	30 %	100 %	10 %	60 %	100 %	57 %	68 %	574.2	00:10:59	00:06:02
28	70 %	70 %	10 %	100 %	30 %	70 %	70 %	67 %	68 %	613	00:07:02	00:04:02
29	70 %	70 %	20 %	100 %	40 %	90 %	100 %	77 %	83 %	504.3	00:08:31	00:04:03
30	70 %	70 %	30 %	100 %	30 %	80 %	100 %	70 %	78 %	428.4	00:09:55	00:04:02
31	70 %	80 %	10 %	100 %	20 %	80 %	70 %	67 %	68 %	426.6	00:07:28	00:02:59
32	70 %	80 %	20 %	100 %	0 %	100 %	90 %	67 %	73 %	511.5	00:08:59	00:04:21
33	70 %	80 %	30 %	100 %	10 %	80 %	90 %	63 %	70 %	454.1	00:11:19	00:04:56
34	70 %	90 %	10 %	100 %	10 %	80 %	90 %	63 %	70 %	525.4	00:07:46	00:03:51
35	70 %	90 %	20 %	100 %	10 %	80 %	100 %	63 %	73 %	611.4	00:09:06	00:05:17
36	70 %	90 %	30 %	100 %	30 %	90 %	100 %	73 %	80 %	739.2	00:11:26	00:08:13
37	80 %	70 %	10 %	100 %	0 %	80 %	100 %	60 %	70 %	493.4	00:06:58	00:03:12
38	80 %	70 %	20 %	100 %	10 %	70 %	90 %	60 %	68 %	586.6	00:08:46	00:04:52
39	80 %	70 %	30 %	100 %	20 %	70 %	90 %	63 %	70 %	457.1	00:10:07	00:04:25
40	80 %	80 %	10 %	100 %	30 %	70 %	80 %	67 %	70 %	479.1	00:07:26	00:03:20
41	80 %	80 %	20 %	100 %	40 %	80 %	70 %	73 %	73 %	549.9	00:09:08	00:04:46
42	80 %	80 %	30 %	100 %	20 %	90 %	100 %	70 %	78 %	568.7	00:10:57	00:05:57
43	80 %	90 %	10 %	100 %	10 %	60 %	90 %	57 %	65 %	340.8	00:08:11	00:02:37
44	80 %	90 %	20 %	100 %	30 %	90 %	80 %	73 %	75 %	515.9	00:09:33	00:04:42

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
45	80 %	90 %	30 %	100 %	10 %	90 %	90 %	67 %	73 %	519.7	00:11:00	00:05:28
46	90 %	70 %	10 %	100 %	10 %	80 %	70 %	63 %	65 %	463.4	00:07:07	00:03:05
47	90 %	70 %	20 %	100 %	30 %	60 %	100 %	63 %	73 %	658.2	00:09:01	00:05:37
48	90 %	70 %	30 %	100 %	10 %	80 %	80 %	63 %	68 %	421.2	00:10:36	00:04:16
49	90 %	80 %	10 %	100 %	30 %	50 %	80 %	60 %	65 %	533.3	00:07:38	00:03:49
50	90 %	80 %	20 %	100 %	20 %	100 %	90 %	73 %	78 %	395.8	00:08:47	00:03:17
51	90 %	80 %	30 %	100 %	40 %	100 %	90 %	80 %	83 %	273.3	00:11:16	00:02:57
52	90 %	90 %	10 %	100 %	30 %	80 %	90 %	70 %	75 %	595.1	00:08:09	00:04:37
53	90 %	90 %	20 %	100 %	40 %	90 %	80 %	77 %	78 %	644.2	00:09:33	00:05:52
54	90 %	90 %	30 %	100 %	10 %	80 %	100 %	63 %	73 %	567.1	00:11:15	00:06:07

B.2. Bosque de bambú de Sagano

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	517.7	00:24:19	00:11:42
2	10 %	70 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	359.1	00:29:48	00:10:05
3	10 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	448.6	00:34:50	00:14:50
4	10 %	80 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	416.5	00:26:05	00:10:08
5	10 %	80 %	20 %	100 %	100 %	100 %	90 %	100 %	98 %	482.5	00:32:45	00:14:59
6	10 %	80 %	30 %	100 %	100 %	100 %	70 %	100 %	93 %	483	00:35:06	00:16:06
7	10 %	90 %	10 %	100 %	100 %	100 %	70 %	100 %	93 %	558.8	00:28:08	00:14:47
8	10 %	90 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	592	00:33:49	00:19:00
9	10 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	425	00:39:18	00:15:59
10	20 %	70 %	10 %	100 %	100 %	100 %	60 %	100 %	90 %	534.2	00:11:21	00:05:48
11	20 %	70 %	20 %	100 %	100 %	100 %	70 %	100 %	93 %	245.6	00:35:21	00:12:21
12	20 %	70 %	30 %	100 %	100 %	100 %	70 %	100 %	93 %	507.1	00:37:14	00:18:01

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
13	20 %	80 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	573.1	00:26:06	00:13:59
14	20 %	80 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	587.4	00:32:34	00:18:08
15	20 %	80 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	504.4	00:38:16	00:18:24
16	20 %	90 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	515.6	00:28:42	00:13:51
17	20 %	90 %	20 %	100 %	100 %	100 %	60 %	100 %	90 %	551	00:34:13	00:17:57
18	20 %	90 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	528.8	00:39:48	00:20:07
19	30 %	70 %	10 %	100 %	100 %	100 %	60 %	100 %	90 %	438.6	00:25:12	00:10:18
20	30 %	70 %	20 %	100 %	100 %	100 %	70 %	100 %	93 %	495.8	00:32:28	00:15:13
21	30 %	70 %	30 %	100 %	100 %	100 %	90 %	100 %	98 %	483.6	00:35:49	00:16:27
22	30 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	554.7	00:27:18	00:14:11
23	30 %	80 %	20 %	100 %	100 %	100 %	90 %	100 %	98 %	517.9	00:32:19	00:15:50
24	30 %	80 %	30 %	100 %	100 %	100 %	60 %	100 %	90 %	574.5	00:37:54	00:20:48
25	30 %	90 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	433.9	00:28:42	00:11:41
26	30 %	90 %	20 %	100 %	100 %	100 %	90 %	100 %	98 %	566	00:35:03	00:18:52
27	30 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	465.1	00:39:11	00:18:00
28	70 %	70 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	401.4	00:23:50	00:08:55
29	70 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	593.7	00:30:50	00:17:19
30	70 %	70 %	30 %	100 %	100 %	100 %	70 %	100 %	93 %	412.9	00:34:31	00:13:35
31	70 %	80 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	451.2	00:25:06	00:10:36
32	70 %	80 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	592.7	00:30:22	00:17:02
33	70 %	80 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	393.2	00:38:20	00:14:25
34	70 %	90 %	10 %	100 %	100 %	100 %	70 %	100 %	93 %	475.6	00:28:02	00:12:27
35	70 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	459	00:33:18	00:14:31
36	70 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	467.3	00:44:00	00:17:17
37	80 %	70 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	526.9	00:24:03	00:11:44
38	80 %	70 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	454.7	00:30:04	00:12:50
39	80 %	70 %	30 %	100 %	100 %	100 %	90 %	100 %	98 %	493.3	00:35:30	00:16:41
40	80 %	80 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	512.5	00:24:40	00:11:47
41	80 %	80 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	575.6	00:31:16	00:17:04

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
42	80 %	80 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	478	00:40:27	00:16:17
43	80 %	90 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	417.6	00:29:18	00:11:31
44	80 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	376.9	00:35:41	00:12:49
45	80 %	90 %	30 %	100 %	100 %	100 %	90 %	100 %	98 %	422.6	00:38:29	00:15:31
46	90 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	477.3	00:24:54	00:11:03
47	90 %	70 %	20 %	100 %	100 %	100 %	90 %	100 %	98 %	585.3	00:30:24	00:16:49
48	90 %	70 %	30 %	100 %	100 %	100 %	90 %	100 %	98 %	408.5	00:36:38	00:14:16
49	90 %	80 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	477.1	00:27:04	00:12:05
50	90 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	446.1	00:31:38	00:13:20
51	90 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	480.9	00:38:16	00:17:35
52	90 %	90 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	663.7	00:27:53	00:17:21
53	90 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	459.3	00:33:25	00:14:33
54	90 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	501.4	00:41:35	00:20:25

B.3. Marilyn Monroe

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	100 %	80 %	100 %	100 %	93 %	95 %	452.7	00:03:19	00:01:25
2	10 %	70 %	20 %	100 %	60 %	100 %	100 %	87 %	90 %	522.3	00:04:02	00:02:00
3	10 %	70 %	30 %	100 %	90 %	100 %	100 %	97 %	98 %	601.4	00:04:53	00:02:49
4	10 %	80 %	10 %	100 %	70 %	100 %	80 %	90 %	88 %	580.8	00:03:28	00:01:53
5	10 %	80 %	20 %	100 %	40 %	100 %	100 %	80 %	85 %	537	00:04:24	00:02:15
6	10 %	80 %	30 %	100 %	70 %	100 %	100 %	90 %	93 %	691.1	00:04:57	00:03:17
7	10 %	90 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	442.4	00:03:48	00:01:35
8	10 %	90 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	576.3	00:05:09	00:02:51
9	10 %	90 %	30 %	100 %	60 %	100 %	100 %	87 %	90 %	592.6	00:05:05	00:02:53
10	20 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	450.2	00:03:16	00:01:23

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
11	20 %	70 %	20 %	100 %	90 %	100 %	90 %	97 %	95 %	577.3	00:04:13	00:02:19
12	20 %	70 %	30 %	100 %	60 %	100 %	100 %	87 %	90 %	500.3	00:04:53	00:02:20
13	20 %	80 %	10 %	100 %	20 %	100 %	100 %	73 %	80 %	617.8	00:03:40	00:02:08
14	20 %	80 %	20 %	100 %	70 %	100 %	90 %	90 %	90 %	491	00:04:21	00:02:02
15	20 %	80 %	30 %	100 %	70 %	100 %	100 %	90 %	93 %	438.8	00:05:18	00:02:14
16	20 %	90 %	10 %	100 %	90 %	100 %	100 %	97 %	98 %	582.3	00:03:49	00:02:06
17	20 %	90 %	20 %	100 %	50 %	100 %	100 %	83 %	88 %	438	00:04:40	00:01:57
18	20 %	90 %	30 %	100 %	70 %	100 %	100 %	90 %	93 %	598	00:05:15	00:03:01
19	30 %	70 %	10 %	100 %	60 %	100 %	90 %	87 %	88 %	354.5	00:03:23	00:01:07
20	30 %	70 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	693.4	00:04:04	00:02:41
21	30 %	70 %	30 %	100 %	70 %	100 %	100 %	90 %	93 %	489.9	00:06:01	00:02:51
22	30 %	80 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	676.4	00:03:48	00:02:26
23	30 %	80 %	20 %	100 %	70 %	100 %	100 %	90 %	93 %	403.3	00:04:34	00:01:45
24	30 %	80 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	633.7	00:05:23	00:03:16
25	30 %	90 %	10 %	100 %	60 %	100 %	100 %	87 %	90 %	407.1	00:04:02	00:01:33
26	30 %	90 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	557.1	00:04:48	00:02:33
27	30 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	210.2	00:05:18	00:01:04
28	70 %	70 %	10 %	100 %	60 %	100 %	100 %	87 %	90 %	522.6	00:03:25	00:01:40
29	70 %	70 %	20 %	100 %	60 %	100 %	100 %	87 %	90 %	475.9	00:04:10	00:01:53
30	70 %	70 %	30 %	100 %	50 %	100 %	90 %	83 %	85 %	516.6	00:04:47	00:02:22
31	70 %	80 %	10 %	100 %	60 %	100 %	100 %	87 %	90 %	579.9	00:03:52	00:02:07
32	70 %	80 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	775.7	00:04:37	00:03:26
33	70 %	80 %	30 %	100 %	60 %	100 %	100 %	87 %	90 %	569.3	00:04:57	00:02:43
34	70 %	90 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	447.4	00:04:10	00:01:46
35	70 %	90 %	20 %	100 %	90 %	100 %	90 %	97 %	95 %	380.3	00:04:49	00:01:45
36	70 %	90 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	416.2	00:05:26	00:02:11
37	80 %	70 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	481.1	00:03:19	00:01:30
38	80 %	70 %	20 %	100 %	70 %	100 %	100 %	90 %	93 %	447	00:04:01	00:01:42
39	80 %	70 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	496.5	00:04:52	00:02:19

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
40	80 %	80 %	10 %	100 %	90 %	100 %	100 %	97 %	98 %	413.4	00:03:38	00:01:25
41	80 %	80 %	20 %	100 %	70 %	100 %	100 %	90 %	93 %	427.6	00:04:07	00:01:40
42	80 %	80 %	30 %	100 %	60 %	100 %	100 %	87 %	90 %	323.8	00:05:16	00:01:38
43	80 %	90 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	599.7	00:04:06	00:02:20
44	80 %	90 %	20 %	100 %	70 %	100 %	100 %	90 %	93 %	485.9	00:04:33	00:02:07
45	80 %	90 %	30 %	100 %	90 %	100 %	90 %	97 %	95 %	763.2	00:05:12	00:03:49
46	90 %	70 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	505.3	00:03:18	00:01:34
47	90 %	70 %	20 %	100 %	40 %	100 %	100 %	80 %	85 %	427.2	00:04:04	00:01:39
48	90 %	70 %	30 %	100 %	70 %	100 %	100 %	90 %	93 %	495.5	00:04:49	00:02:17
49	90 %	80 %	10 %	100 %	70 %	100 %	90 %	90 %	90 %	455.6	00:03:32	00:01:31
50	90 %	80 %	20 %	100 %	60 %	100 %	90 %	87 %	88 %	465	00:04:24	00:01:57
51	90 %	80 %	30 %	100 %	70 %	100 %	90 %	90 %	90 %	416.5	00:05:06	00:02:02
52	90 %	90 %	10 %	100 %	50 %	100 %	100 %	83 %	88 %	664.3	00:03:44	00:02:20
53	90 %	90 %	20 %	100 %	80 %	100 %	100 %	93 %	95 %	562.5	00:04:23	00:02:21
54	90 %	90 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	716.6	00:05:12	00:03:34

B.4. Imagen 100x100

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	100 %	100 %	70 %	60 %	90 %	83 %	566.4	00:00:17	00:00:09
2	10 %	70 %	20 %	100 %	100 %	80 %	50 %	93 %	83 %	467.7	00:00:21	00:00:09
3	10 %	70 %	30 %	100 %	100 %	100 %	60 %	100 %	90 %	449	00:00:25	00:00:11
4	10 %	80 %	10 %	100 %	100 %	90 %	30 %	97 %	80 %	373.3	00:00:18	00:00:07
5	10 %	80 %	20 %	100 %	100 %	80 %	40 %	93 %	80 %	405.8	00:00:21	00:00:08
6	10 %	80 %	30 %	100 %	100 %	80 %	30 %	93 %	78 %	285.7	00:00:27	00:00:07
7	10 %	90 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	592.3	00:00:18	00:00:10
8	10 %	90 %	20 %	100 %	100 %	80 %	20 %	93 %	75 %	573.1	00:00:23	00:00:13

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
9	10 %	90 %	30 %	100 %	100 %	100 %	40 %	100 %	85 %	436.4	00:00:26	00:00:11
10	20 %	70 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	623	00:00:17	00:00:10
11	20 %	70 %	20 %	100 %	100 %	60 %	40 %	87 %	75 %	572.3	00:00:20	00:00:11
12	20 %	70 %	30 %	100 %	100 %	90 %	50 %	97 %	85 %	379.8	00:00:24	00:00:09
13	20 %	80 %	10 %	100 %	100 %	60 %	40 %	87 %	75 %	512.6	00:00:18	00:00:09
14	20 %	80 %	20 %	100 %	100 %	90 %	50 %	97 %	85 %	532.1	00:00:22	00:00:11
15	20 %	80 %	30 %	100 %	90 %	90 %	70 %	93 %	88 %	491.3	00:00:25	00:00:12
16	20 %	90 %	10 %	100 %	100 %	90 %	70 %	97 %	90 %	526.1	00:00:19	00:00:09
17	20 %	90 %	20 %	100 %	100 %	100 %	30 %	100 %	83 %	661	00:00:22	00:00:14
18	20 %	90 %	30 %	100 %	100 %	80 %	40 %	93 %	80 %	395.2	00:00:26	00:00:10
19	30 %	70 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	622.9	00:00:17	00:00:10
20	30 %	70 %	20 %	100 %	100 %	90 %	30 %	97 %	80 %	373.6	00:00:21	00:00:08
21	30 %	70 %	30 %	100 %	100 %	100 %	50 %	100 %	88 %	515	00:00:24	00:00:12
22	30 %	80 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	381.7	00:00:17	00:00:06
23	30 %	80 %	20 %	100 %	100 %	100 %	60 %	100 %	90 %	457.8	00:00:23	00:00:10
24	30 %	80 %	30 %	100 %	100 %	90 %	50 %	97 %	85 %	449.5	00:00:25	00:00:11
25	30 %	90 %	10 %	100 %	100 %	70 %	40 %	90 %	78 %	409.1	00:00:18	00:00:07
26	30 %	90 %	20 %	100 %	100 %	90 %	70 %	97 %	90 %	517	00:00:22	00:00:11
27	30 %	90 %	30 %	100 %	100 %	90 %	50 %	97 %	85 %	572.5	00:00:26	00:00:14
28	70 %	70 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	595.7	00:00:18	00:00:10
29	70 %	70 %	20 %	100 %	90 %	60 %	40 %	83 %	73 %	366.9	00:00:21	00:00:07
30	70 %	70 %	30 %	100 %	100 %	80 %	70 %	93 %	88 %	572.6	00:00:23	00:00:13
31	70 %	80 %	10 %	100 %	100 %	100 %	60 %	100 %	90 %	532.5	00:00:18	00:00:09
32	70 %	80 %	20 %	100 %	100 %	90 %	70 %	97 %	90 %	531.5	00:00:22	00:00:11
33	70 %	80 %	30 %	100 %	100 %	90 %	10 %	97 %	75 %	507.8	00:00:24	00:00:12
34	70 %	90 %	10 %	100 %	100 %	90 %	40 %	97 %	83 %	532	00:00:18	00:00:09
35	70 %	90 %	20 %	100 %	100 %	80 %	70 %	93 %	88 %	436.8	00:00:22	00:00:09
36	70 %	90 %	30 %	100 %	100 %	80 %	60 %	93 %	85 %	397.5	00:00:27	00:00:10
37	80 %	70 %	10 %	100 %	100 %	90 %	50 %	97 %	85 %	442.3	00:00:18	00:00:07

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
38	80 %	70 %	20 %	100 %	100 %	60 %	30 %	87 %	73 %	386.9	00:00:20	00:00:08
39	80 %	70 %	30 %	100 %	100 %	80 %	50 %	93 %	83 %	432.4	00:00:25	00:00:10
40	80 %	80 %	10 %	100 %	100 %	80 %	40 %	93 %	80 %	570.9	00:00:19	00:00:10
41	80 %	80 %	20 %	100 %	100 %	90 %	20 %	97 %	78 %	499.9	00:00:21	00:00:10
42	80 %	80 %	30 %	100 %	100 %	90 %	60 %	97 %	88 %	287.6	00:00:26	00:00:07
43	80 %	90 %	10 %	100 %	100 %	100 %	70 %	100 %	93 %	512.4	00:00:19	00:00:09
44	80 %	90 %	20 %	100 %	100 %	90 %	60 %	97 %	88 %	453.6	00:00:22	00:00:10
45	80 %	90 %	30 %	100 %	100 %	80 %	50 %	93 %	83 %	523.6	00:00:26	00:00:13
46	90 %	70 %	10 %	100 %	100 %	70 %	50 %	90 %	80 %	448.5	00:00:17	00:00:07
47	90 %	70 %	20 %	100 %	100 %	80 %	80 %	93 %	90 %	528.1	00:00:22	00:00:11
48	90 %	70 %	30 %	100 %	100 %	100 %	30 %	100 %	83 %	358.6	00:00:24	00:00:08
49	90 %	80 %	10 %	100 %	100 %	60 %	70 %	87 %	83 %	601.7	00:00:18	00:00:10
50	90 %	80 %	20 %	100 %	100 %	60 %	40 %	87 %	75 %	621.1	00:00:22	00:00:13
51	90 %	80 %	30 %	100 %	90 %	80 %	50 %	90 %	80 %	646.1	00:00:25	00:00:15
52	90 %	90 %	10 %	100 %	100 %	90 %	70 %	97 %	90 %	538.6	00:00:19	00:00:10
53	90 %	90 %	20 %	100 %	100 %	50 %	50 %	83 %	75 %	464.2	00:00:23	00:00:11
54	90 %	90 %	30 %	100 %	100 %	90 %	40 %	97 %	83 %	539.7	00:00:26	00:00:14

B.5. Muerte de un miliciano

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	524.4	00:15:21	00:07:32
2	10 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	360.8	00:18:25	00:06:16
3	10 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	608.1	00:21:49	00:12:37
4	10 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	646.2	00:15:53	00:09:34
5	10 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	74	00:20:01	00:01:21
6	10 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	531.2	00:22:42	00:11:28

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
7	10 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	511	00:16:54	00:08:07
8	10 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	345.5	00:21:26	00:07:04
9	10 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	468.8	00:23:57	00:10:45
10	20 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	679.1	00:15:11	00:09:38
11	20 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	536.2	00:18:52	00:09:31
12	20 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	421.3	00:21:11	00:08:30
13	20 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	357.2	00:15:40	00:05:14
14	20 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	590.7	00:20:30	00:11:30
15	20 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	540.3	00:21:38	00:11:10
16	20 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	532.8	00:15:59	00:07:59
17	20 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	506.4	00:27:16	00:09:55
18	20 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	586.7	00:24:17	00:13:40
19	30 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	607.7	00:14:46	00:08:23
20	30 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	495.3	00:17:29	00:08:10
21	30 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	436.7	00:21:35	00:09:00
22	30 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	287.3	00:15:52	00:04:16
23	30 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	410.4	00:20:02	00:07:47
24	30 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	524.7	00:22:44	00:11:24
25	30 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	505.8	00:16:11	00:07:40
26	30 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	500.7	00:19:24	00:09:13
27	30 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	629.7	00:25:31	00:15:25
28	70 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	473.7	00:14:51	00:06:32
29	70 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	526.1	00:18:52	00:09:23
30	70 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	619.5	00:20:13	00:11:55
31	70 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	569.1	00:16:06	00:08:37
32	70 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	469.3	00:18:30	00:08:13
33	70 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	421	00:22:28	00:09:02
34	70 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	556.8	00:16:14	00:08:31
35	70 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	453.1	00:19:45	00:08:29

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
36	70 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	542	00:23:41	00:12:17
37	80 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	644.1	00:14:38	00:08:46
38	80 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	444.8	00:17:49	00:07:29
39	80 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	475	00:22:05	00:09:59
40	80 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	595.4	00:15:39	00:08:43
41	80 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	513.9	00:19:51	00:09:40
42	80 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	562.1	00:31:36	00:12:11
43	80 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	360.7	00:16:13	00:05:29
44	80 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	663.8	00:20:38	00:12:58
45	80 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	467.5	00:23:14	00:10:21
46	90 %	70 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	475.8	00:16:45	00:07:30
47	90 %	70 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	455.5	00:20:12	00:08:34
48	90 %	70 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	665.6	00:23:10	00:14:48
49	90 %	80 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	598.7	00:16:39	00:09:21
50	90 %	80 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	459.9	00:18:52	00:08:13
51	90 %	80 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	449.9	00:22:15	00:09:33
52	90 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	486.7	00:16:47	00:07:41
53	90 %	90 %	20 %	100 %	100 %	100 %	100 %	100 %	100 %	611	00:20:06	00:11:38
54	90 %	90 %	30 %	100 %	100 %	100 %	100 %	100 %	100 %	522.7	00:23:41	00:11:52

B.6. Hojas en el parque nacional Glacier

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
1	10 %	70 %	10 %	90 %	100 %	100 %	90 %	97 %	95 %	524.8	00:07:54	00:03:52
2	10 %	70 %	20 %	90 %	90 %	100 %	90 %	93 %	93 %	433.4	00:09:39	00:03:57
3	10 %	70 %	30 %	100 %	80 %	100 %	100 %	93 %	95 %	484.1	00:11:36	00:05:22
4	10 %	80 %	10 %	100 %	70 %	100 %	100 %	90 %	93 %	567.2	00:08:35	00:04:34

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
5	10 %	80 %	20 %	100 %	60 %	100 %	70 %	87 %	83 %	490.7	00:10:58	00:05:06
6	10 %	80 %	30 %	100 %	70 %	100 %	80 %	90 %	88 %	563.2	00:12:37	00:06:48
7	10 %	90 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	626.4	00:09:14	00:05:28
8	10 %	90 %	20 %	100 %	70 %	100 %	90 %	90 %	90 %	524.6	00:10:49	00:05:23
9	10 %	90 %	30 %	90 %	90 %	100 %	80 %	93 %	90 %	552.3	00:12:56	00:06:50
10	20 %	70 %	10 %	100 %	60 %	100 %	80 %	87 %	85 %	626.9	00:08:16	00:04:49
11	20 %	70 %	20 %	100 %	90 %	100 %	90 %	97 %	95 %	541.6	00:09:46	00:05:00
12	20 %	70 %	30 %	90 %	90 %	100 %	90 %	93 %	93 %	472.5	00:11:59	00:05:24
13	20 %	80 %	10 %	90 %	100 %	100 %	90 %	97 %	95 %	626.3	00:09:00	00:05:17
14	20 %	80 %	20 %	90 %	100 %	100 %	80 %	97 %	93 %	568.7	00:10:05	00:05:27
15	20 %	80 %	30 %	90 %	80 %	100 %	70 %	90 %	85 %	636.4	00:12:28	00:07:35
16	20 %	90 %	10 %	100 %	90 %	100 %	80 %	97 %	93 %	487.1	00:08:53	00:04:04
17	20 %	90 %	20 %	100 %	80 %	100 %	90 %	93 %	93 %	486.5	00:11:07	00:05:11
18	20 %	90 %	30 %	100 %	100 %	100 %	80 %	100 %	95 %	554.8	00:13:21	00:07:06
19	30 %	70 %	10 %	100 %	90 %	100 %	80 %	97 %	93 %	383.4	00:07:59	00:02:50
20	30 %	70 %	20 %	100 %	70 %	100 %	80 %	90 %	88 %	426	00:10:06	00:04:04
21	30 %	70 %	30 %	100 %	90 %	100 %	70 %	97 %	90 %	568.2	00:11:39	00:06:20
22	30 %	80 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	520.1	00:08:42	00:04:15
23	30 %	80 %	20 %	100 %	90 %	100 %	80 %	97 %	93 %	479.9	00:10:10	00:04:37
24	30 %	80 %	30 %	90 %	100 %	100 %	90 %	97 %	95 %	570.4	00:12:01	00:06:33
25	30 %	90 %	10 %	90 %	70 %	100 %	80 %	87 %	85 %	463.3	00:08:47	00:03:51
26	30 %	90 %	20 %	90 %	70 %	100 %	100 %	87 %	90 %	572.8	00:11:07	00:06:02
27	30 %	90 %	30 %	100 %	70 %	100 %	80 %	90 %	88 %	566.5	00:12:21	00:06:42
28	70 %	70 %	10 %	90 %	80 %	100 %	90 %	90 %	90 %	536.3	00:08:26	00:04:14
29	70 %	70 %	20 %	90 %	90 %	100 %	60 %	93 %	85 %	392.7	00:09:42	00:03:36
30	70 %	70 %	30 %	100 %	90 %	100 %	70 %	97 %	90 %	541.6	00:11:16	00:05:46
31	70 %	80 %	10 %	100 %	60 %	100 %	90 %	87 %	88 %	648	00:08:18	00:05:02
32	70 %	80 %	20 %	100 %	100 %	100 %	70 %	100 %	93 %	508.9	00:10:22	00:04:59
33	70 %	80 %	30 %	90 %	80 %	100 %	90 %	90 %	90 %	706	00:11:34	00:07:47

N.	Pch	Pc	Pm	Diagonal	Horizontal	Vertical	Imagen	Pixeles	General	Generación	Total	Promedio
34	70 %	90 %	10 %	100 %	100 %	100 %	80 %	100 %	95 %	421.5	00:09:29	00:03:47
35	70 %	90 %	20 %	100 %	80 %	100 %	80 %	93 %	90 %	556.5	00:10:42	00:05:40
36	70 %	90 %	30 %	100 %	60 %	100 %	80 %	87 %	85 %	424	00:12:37	00:05:08
37	80 %	70 %	10 %	100 %	100 %	100 %	90 %	100 %	98 %	595.5	00:08:10	00:04:33
38	80 %	70 %	20 %	90 %	90 %	100 %	100 %	93 %	95 %	571.3	00:09:32	00:05:08
39	80 %	70 %	30 %	100 %	70 %	100 %	90 %	90 %	90 %	576.4	00:11:27	00:06:18
40	80 %	80 %	10 %	100 %	70 %	100 %	90 %	90 %	90 %	428	00:08:25	00:03:22
41	80 %	80 %	20 %	100 %	80 %	100 %	70 %	93 %	88 %	574.6	00:10:22	00:05:39
42	80 %	80 %	30 %	100 %	80 %	100 %	80 %	93 %	90 %	371.8	00:11:56	00:04:15
43	80 %	90 %	10 %	90 %	70 %	100 %	100 %	87 %	90 %	500.7	00:08:49	00:04:09
44	80 %	90 %	20 %	100 %	70 %	100 %	90 %	90 %	90 %	494.5	00:10:23	00:04:53
45	80 %	90 %	30 %	100 %	60 %	100 %	90 %	87 %	88 %	456.2	00:13:02	00:05:40
46	90 %	70 %	10 %	100 %	60 %	100 %	60 %	87 %	80 %	597.9	00:08:03	00:04:29
47	90 %	70 %	20 %	100 %	90 %	100 %	100 %	97 %	98 %	631.1	00:10:24	00:06:14
48	90 %	70 %	30 %	90 %	100 %	100 %	90 %	97 %	95 %	402.6	00:11:22	00:04:22
49	90 %	80 %	10 %	100 %	80 %	100 %	80 %	93 %	90 %	722.8	00:08:56	00:06:03
50	90 %	80 %	20 %	90 %	60 %	100 %	80 %	83 %	83 %	523.3	00:10:35	00:05:15
51	90 %	80 %	30 %	100 %	80 %	100 %	90 %	93 %	93 %	471.5	00:12:18	00:05:33
52	90 %	90 %	10 %	100 %	90 %	100 %	60 %	97 %	88 %	361.4	00:08:54	00:03:00
53	90 %	90 %	20 %	100 %	100 %	100 %	80 %	100 %	95 %	424.2	00:10:58	00:04:19
54	90 %	90 %	30 %	90 %	70 %	100 %	80 %	87 %	85 %	397.2	00:12:26	00:04:43

Bibliografía

Biology Notes for IGCSE 2014 (2014). Chromosomes, dna, genes and alleles. [figura]. https://biology-igcse.weebly.com/-chromosomes-dna.html.

Cuba Educa (2014). División celular en eucariotas. [figura]. https://bit.ly/2sSCZYh.

El-Samie, F. E. A., Ahmed, H. E. H., Elashry, I. F., Shahieen, M. H., Faragallah, O. S., El-Rabaie, E.-S. M., and Alshebeili, S. A. (2013). *Image encryption: a communication perspective*. CRC Press.

Enayatifar, R. and Abdullah, A. H. (2011). Image security via genetic algorithm. In *International Conference on Computer and Software Modeling*, volume 14, pages 198–203.

Fotografía de Ansel Easton Adams (1942). Hojas en primer plano en el parque nacional glacier. [figura]. https://es.wikipedia.org/wiki/Ansel_Adams. Citado en Enero 2020.

Fotografía de Cristobal García (2016). Un visitante pasa por delante de la famosa foto del miliciano español en una exposición. [figura]. https://bit.ly/39moqMa. Citado en Enero 2020.

Fotografía de Dwight Hooker (1972). Lenna. [figura]. https://en.wikipedia.org/wiki/Lenna. Citado en Enero 2020.

Fotografía de Eve Arnold (1960). Marilyn monroe en el set "the misfits". [figura]. https://bit.ly/2OKNAMw. Citado en Enero 2020.

Fotografia de Curt Smith (2019). Bosque de bambú de arashiyama (kioto, japón). [figura]. https://bit.ly/2SBt9To. Citado en Febrero 2020.

Gleick, J. (2011). *Chaos: Making a new science*. Open Road Media.

Gonzales, R. C. and Woods, R. E. (2002). *Digital Image Processing. New Jersey: PrenticeHall*. Inc.

Imagen del almbum Una Puntata (2019). 100x100. [figura]. https://music.apple.com/th/artist/100x100/id1475610733. Citado en Febrero 2020.

López, M. J. L. (2011). *Criptografía y seguridad en computadores*. Universidad de Jaén, Versión.

Ogata, K. and Sanchez, G. L. P. (1987). *Dinámica de sistemas*. Prentice-Hall Hispanoamericana.

Singh, S. (2000). *Codigos Secretos*. Debate.

Sivanandam, S. and Deepa, S. (2008). *Introduction To Genetic Algorithm*. Springer.

Tecnical. Código ascii. [figura]. `https://bit.ly/37G3Plz`. Citado en Enero 2020.