Proyecto Parcial 1

Gabriel A Moo Martínez

Axel E Quen Euan

Carlos A Domínguez Borges

ISC, 6to ”A”

Facultad de Ingeniería
Universidad Autónoma de Campeche

Inteligencia Artificial

Dr. José C Aguilar Canepa

16 de marzo de 2025

Introducción

En un mundo cada vez más digitalizado, la seguridad de la información se ha convertido en una prioridad tanto para individuos como para empresas. La protección de datos personales, cuentas bancarias, sistemas gubernamentales y plataformas en línea depende, en gran medida, del uso de contraseñas robustas y difíciles de descifrar. Sin embargo, los métodos tradicionales de generación de contraseñas suelen basarse en reglas predefinidas o en la aleatorización simple de caracteres, lo que no siempre garantiza una combinación óptima entre seguridad y facilidad de uso.

El problema de la seguridad de las contraseñas es crucial en la era digital. Estudios han demostrado que una gran cantidad de usuarios utiliza contraseñas débiles, repetitivas o fácilmente predecibles, como “123456” o “password”, lo que facilita el acceso no autorizado a cuentas y sistemas. Incluso cuando se imponen restricciones en la creación de contraseñas (como la obligación de incluir mayúsculas, números y caracteres especiales), los usuarios tienden a elegir combinaciones predecibles, reduciendo la efectividad de estas medidas.

Los métodos tradicionales para generar contraseñas seguras incluyen la aleatorización total y las reglas basadas en patrones específicos. Sin embargo, estos enfoques pueden no ser los más eficientes, ya que o bien generan contraseñas difíciles de recordar para el usuario o bien producen claves que, aunque cumplen con ciertos requisitos formales, no son lo suficientemente resistentes frente a ataques avanzados.

Aquí es donde los algoritmos genéticos pueden marcar una diferencia. Al tratar la generación de contraseñas como un problema de optimización, es posible encontrar combinaciones de caracteres que maximicen la seguridad sin sacrificar completamente la memorización y usabilidad.

Para abordar este problema, en este trabajo se propone el uso de **algoritmos genéticos** como una alternativa innovadora para la generación de contraseñas seguras. Aplicando principios de evolución y selección natural, estos algoritmos pueden optimizar la generación de claves, asegurando una combinación eficiente entre robustez y usabilidad. A través de este estudio, se busca demostrar cómo los algoritmos genéticos pueden generar contraseñas seguras y difíciles de descifrar, contribuyendo así a mejorar la seguridad digital.

Planteamiento del problema

En la actualidad, la seguridad en línea es un tema de gran importancia, dado que muchas personas utilizan contraseñas débiles o repetitivas, esto lamentablemente, las hace vulnerables a ataques cibernéticos. Esto representa un riesgo significativo para la información personal, financiera e incluso empresarial.

Frente a este problema, surge la necesidad de encontrar un método eficiente para generar contraseñas robustas que sean difíciles de descifrar. Sin embargo, es importante reconocer que las contraseñas generadas automáticamente no siempre son fáciles de recordar. Aquí es donde la inteligencia artificial, específicamente los algoritmos genéticos, puede ofrecer una solución innovadora. Estos algoritmos imitan el proceso de evolución natural para encontrar soluciones óptimas a problemas complejos, como la creación de contraseñas altamente seguras que cumplan con criterios específicos de fortaleza.

El objetivo de este proyecto es demostrar cómo los algoritmos genéticos pueden utilizarse para generar contraseñas robustas y difíciles de descifrar. Aunque estas contraseñas no están diseñadas para ser memorizadas fácilmente, pueden ser almacenadas de manera segura en gestores de contraseñas, lo que permite equilibrar la seguridad con la practicidad. De esta manera, se espera contribuir a mejorar la seguridad digital de los usuarios y las empresas, ofreciendo una alternativa eficaz frente a las contraseñas débiles o repetitivas.

Propuesta de solución

Técnica Escogida: Algoritmos Genéticos

Los algoritmos genéticos son una técnica de optimización que imita el proceso de selección natural y evolución biológica. En este caso, se utilizan para generar contraseñas que cumplan con los siguientes criterios de seguridad:

- Longitud mínima de 12 caracteres.
- Inclusión de letras mayúsculas y minúsculas.
- Uso de números y caracteres especiales.
- Dificultad para ser descifradas mediante ataques de fuerza bruta.

Diseño del modelo

El diseño del modelo se basa en la implementación de un algoritmo genético utilizando la biblioteca DEAP (Distributed Evolutionary Algorithms in Python). A continuación, se describe el proceso:

1.- Representación de la solución (individuos)

Cada individuo (contraseña) se representa como una lista de caracteres. En este caso, cada contraseña tiene una longitud fija de 12 caracteres.

Los caracteres posibles incluyen letras mayúsculas, minúsculas, números y símbolos especiales. (string.ascii_letters + string.digits + string.punctuation).

```
def init_genetic_algorithm(self):
    creator.create("FitnessMax", base.Fitness, weights=(1.0,))
    creator.create("Individual", list, fitness=creator.FitnessMax)

    self.toolbox = base.Toolbox()
    caracteres = string.ascii_letters + string.digits + string.punctuation
    self.toolbox.register("attr_char", random.choice, caracteres)
    self.toolbox.register("individual", tools.initRepeat, creator.Individual, self.toolbox.attr_char, 12)
    self.toolbox.register("population", tools.initRepeat, list, self.toolbox.individual)
```

2.- Función de evaluación

La función de evaluación (evaluar_contraseña) asigna una puntuación a cada contraseña en función de su cumplimiento con los criterios de seguridad:

- Uso de al menos una letra minúscula.
- Uso de al menos una letra mayúscula.
- Uso de al menos un número.
- Uso de al menos un carácter especial.
- Longitud mínima de 12 caracteres.

La puntuación máxima es 5, que indica que la contraseña cumple con todos los criterios.

```
def evaluar_contraseña(ind):
    contraseña = "".join(ind)
    puntuacion = sum([
        any(c.islower() for c in contraseña),
        any(c.isupper() for c in contraseña),
        any(c.isdigit() for c in contraseña),
        any(c in string.punctuation for c in contraseña),
        len(contraseña) >= 12
    ])
    return puntuacion,
```

3.- Operadores genéticos

- Cruzamiento (mate): Se utiliza el método de cruce de dos puntos (cxTwoPoint) para combinar dos individuos y generar descendencia.

- Mutación (mutate): Se aplica una mutación aleatoria a cada carácter del individuo con una probabilidad del 30% (indpb=0.3). Si se produce una mutación, el carácter se reemplaza por otro carácter válido.
- Selección (select): Se utiliza la selección por torneo (selTournament) para elegir a los individuos más aptos de la población.

```
        self.toolbox.register("mate", tools.cxTwoPoint)

        def mutar_contraseña(ind, indpb=0.3):
            for i in range(len(ind)):
                if random.random() < indpb:
                    opciones = list(caracteres)
                    opciones.remove(ind[i])
                    ind[i] = random.choice(opciones)
            return ind,

        self.toolbox.register("mutate", mutar_contraseña)
        self.toolbox.register("select", tools.selTournament, tournsize=3)
```

4.- Evolución de la Población

1. Se genera una población inicial de 100 individuos.
2. El proceso evolutivo se ejecuta durante un máximo de 50 generaciones (max_gen = 50).
3. En cada generación, se aplican los operadores de cruce y mutación para generar una nueva población.
4. El proceso se detiene si se encuentra una contraseña que cumple con todos los criterios de seguridad (puntuación = 5).

```
def generate_password(self):
    poblacion = self.toolbox.population(n=100)
    max_gen = 50
    mu, lambda_ = 100, 200

    for gen in range(max_gen):
        offspring = algorithms.varAnd(poblacion, self.toolbox, cxpb=0.5, mutpb=0.2)
        fits = [self.toolbox.evaluate(ind)[0] for ind in offspring]

        if max(fits) == 5:
            break

        poblacion = self.toolbox.select(offspring, mu)
```

5.- Resultado final

La mejor contraseña generada se selecciona utilizando tools.selBest y se muestra en la interfaz gráfica.

```
mejor_contraseña = tools.selBest(poblacion, 1)[0]
self.password_output.setText("".join(mejor_contraseña))
```

Características del problema/solución.

Problema: Generar contraseñas seguras que cumplan con criterios específicos de robustez.

Solución: Utilizar algoritmos genéticos para evolucionar una población de contraseñas hasta encontrar una que cumpla con todos los requisitos de seguridad.

Ventajas:

- Las contraseñas generadas son altamente seguras y difíciles de descifrar.
- El proceso es automatizado y puede adaptarse a diferentes criterios de seguridad.
- Las contraseñas pueden ser almacenadas en gestores de contraseñas para su uso práctico.

Interfaz Gráfica

La interfaz gráfica, desarrollada con PyQt6, permite a los usuarios generar y guardar contraseñas de manera intuitiva. Las contraseñas generadas se muestran en un campo de texto, y las contraseñas guardadas se almacenan en una tabla para su posterior consulta.

Resultados experimentales

Descripciones de las pruebas realizadas

Se realizaron cinco iteraciones del algoritmo genético con una población inicial de 100 individuos durante 50 generaciones y se evaluó el fitness de cada contraseña en función de su longitud, diversidad de caracteres y resistencia ante ataques de fuerza bruta.

Análisis del fitness

El fitness máximo se estabilizó en valores cercanos a 68-70 después de algunas generaciones, indicando que el algoritmo encuentra soluciones óptimas relativamente rápido. El promedio de fitness mostró una tendencia creciente en todas las pruebas, lo que indica que la calidad general de la población mejoró progresivamente.

Convergencia y Limitaciones

En varias iteraciones, el fitness máximo alcanzó un tope temprano, lo que sugiere convergencia prematura y falta de exploración de soluciones más diversas.

La diferencia entre el máximo y el promedio fitness muestra que todavía hay individuos con contraseñas de baja calidad en la población.

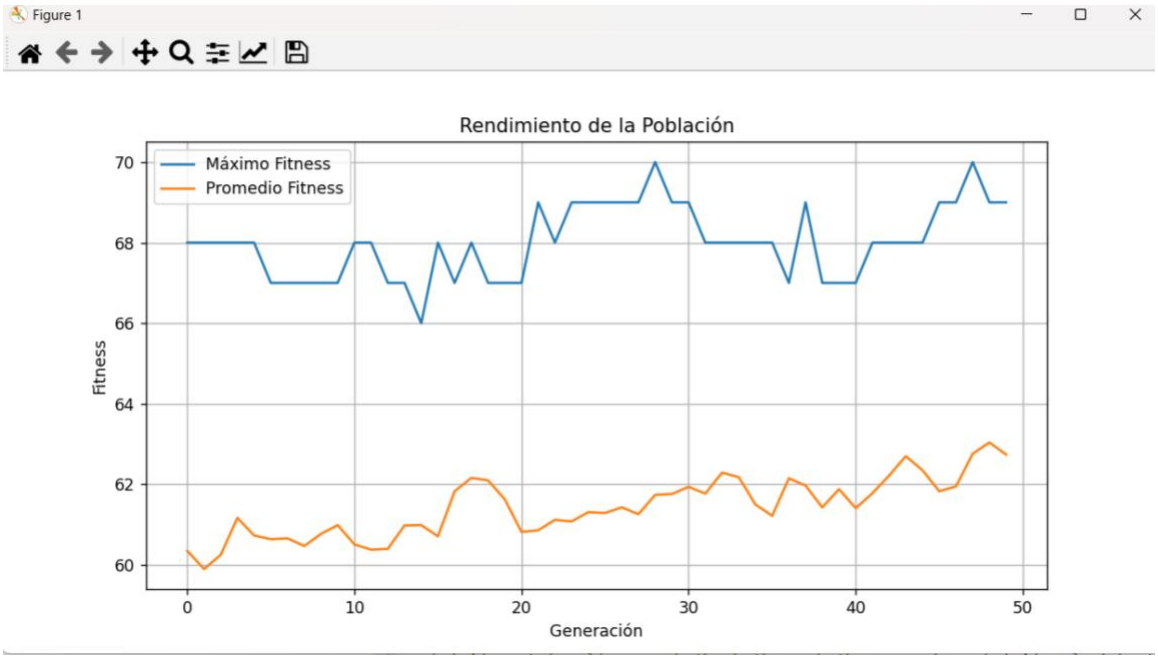
Figure 1
Rendimiento de la Población
Máximo Fitness
Promedio Fitness
Fitness
Generación
70
68
66
64
62
60
0
10
20
30
40
50

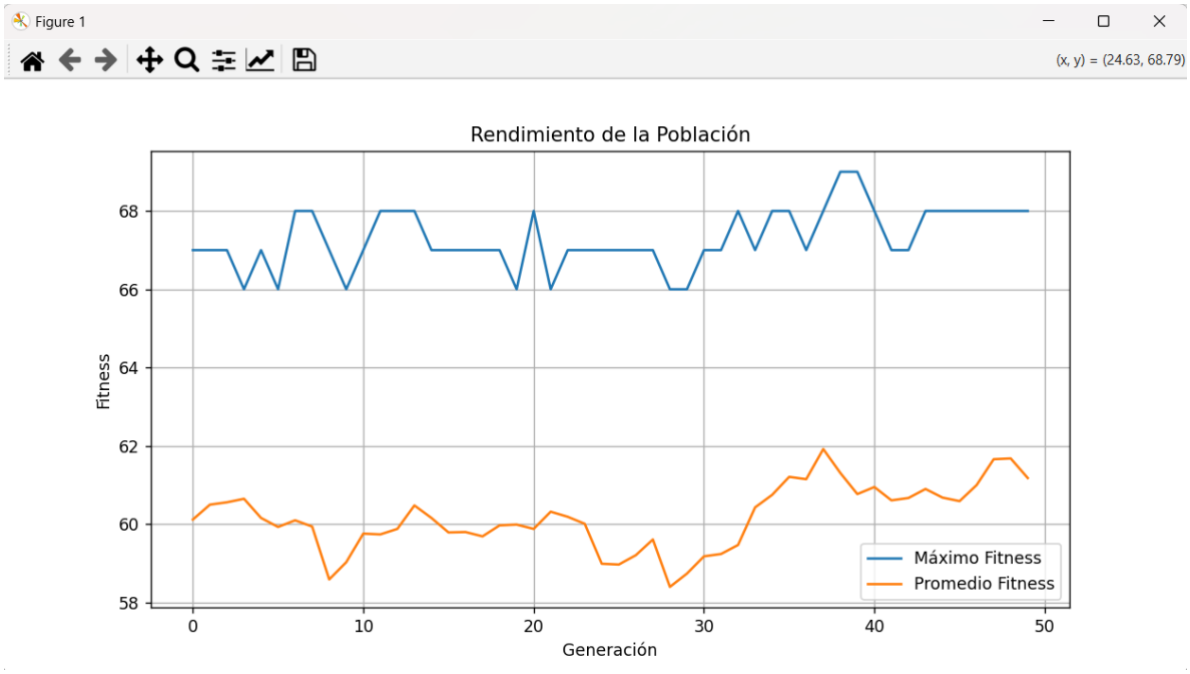
Figure 1
(x, y) = (24.63, 68.79)
Rendimiento de la Población
Máximo Fitness
Promedio Fitness
Fitness
Generación
68
66
64
62
60
58
0
10
20
30
40
50

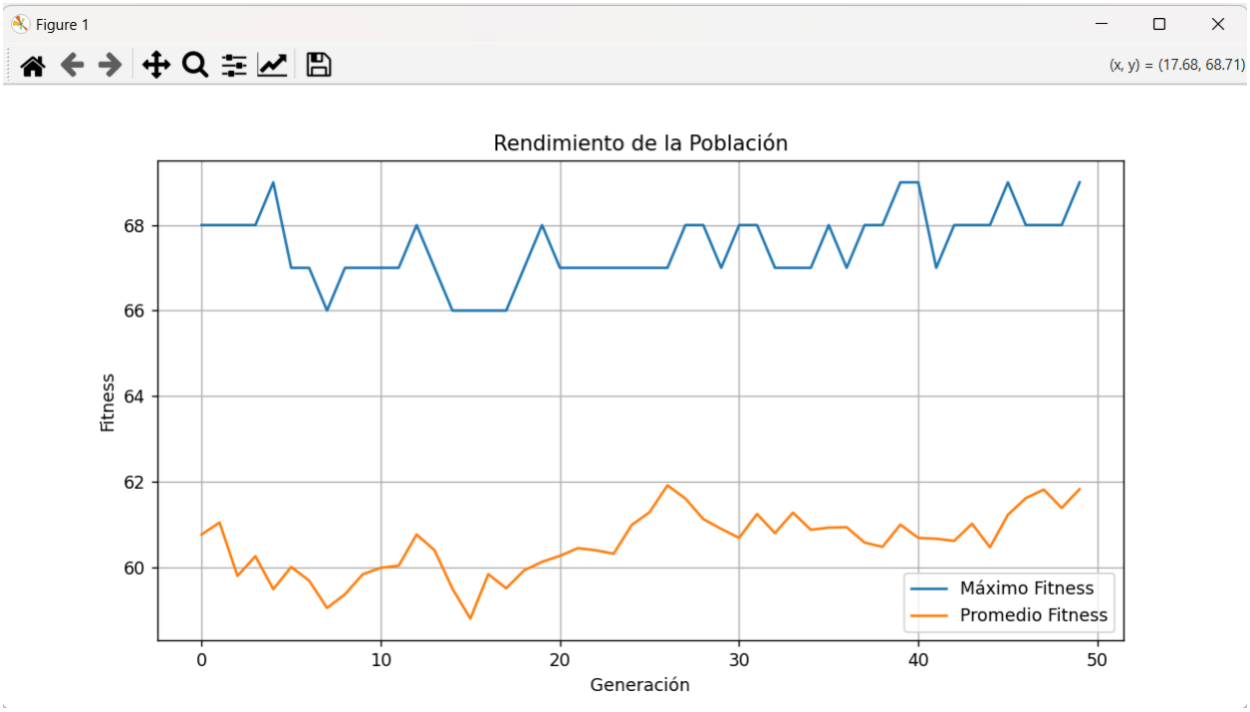
Figure 1
(x, y) = (17.68, 68.71)
Rendimiento de la Población
Fitness
Generación
Máximo Fitness
Promedio Fitness
68
66
64
62
60
0
10
20
30
40
50

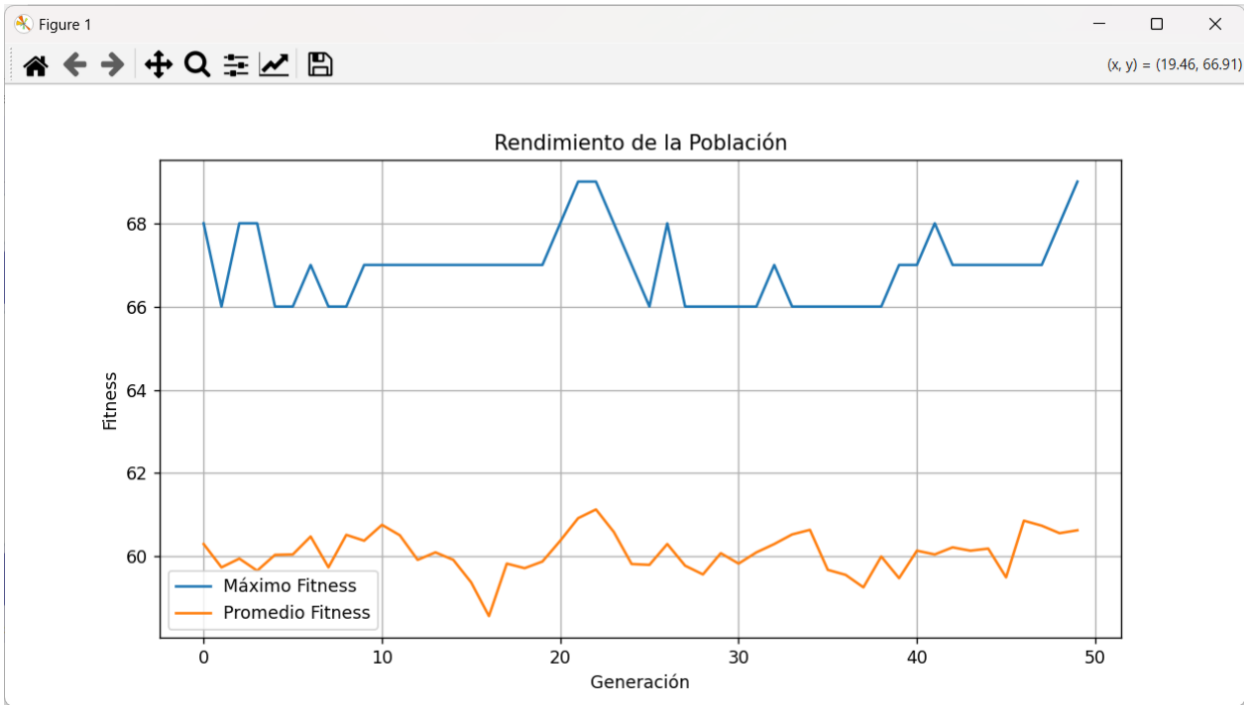
Figure 1
(x, y) = (19.46, 66.91)
Rendimiento de la Población
Fitness
Generación
Máximo Fitness
Promedio Fitness
68
66
64
62
60
0
10
20
30
40
50

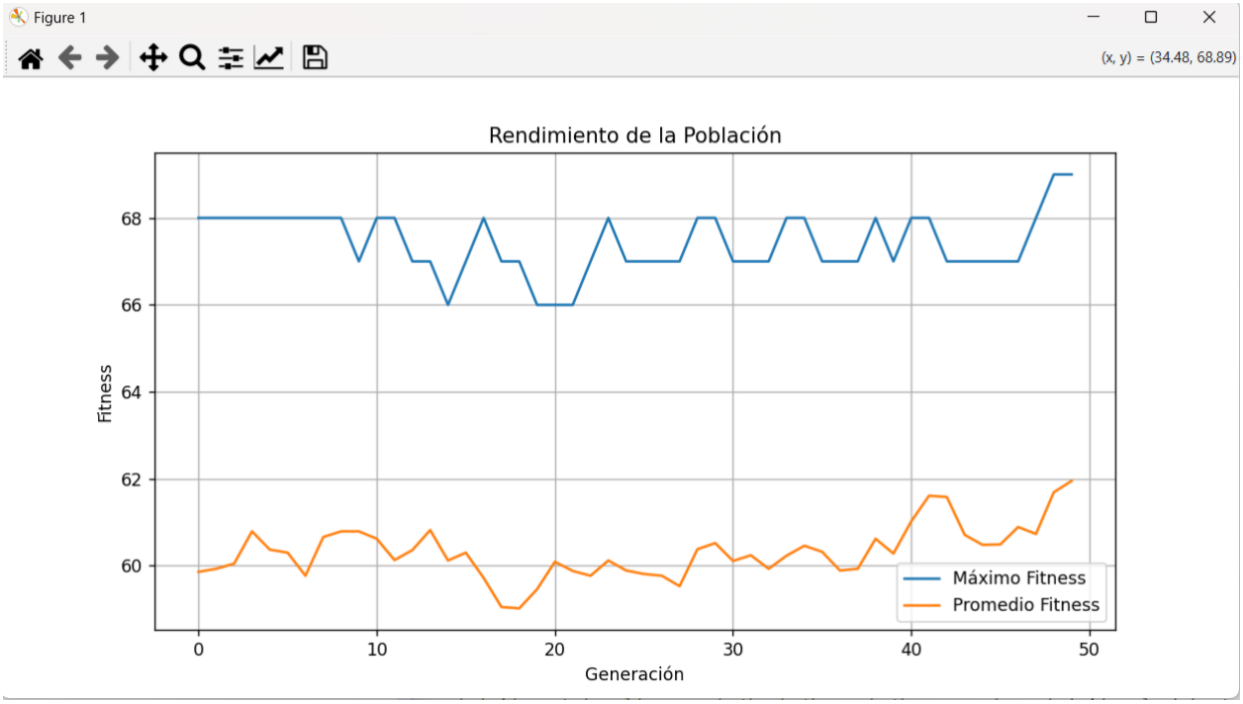
Figure 1
(x, y) = (34.48, 68.89)
Rendimiento de la Población
Fitness
Generación
Máximo Fitness
Promedio Fitness
68
66
64
62
60
0
10
20
30
40
50

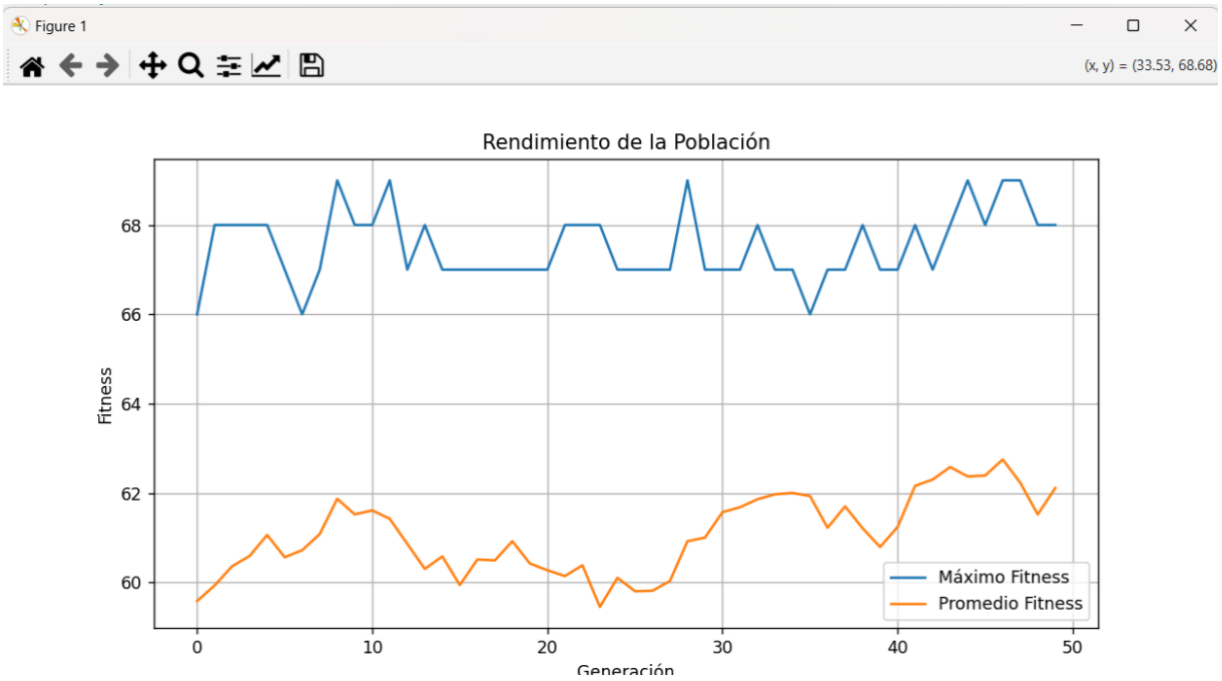
Figure 1
(x, y) = (33.53, 68.68)
Rendimiento de la Población
Fitness
Generación
Máximo Fitness
Promedio Fitness
68
66
64
62
60
0
10
20
30
40
50

Conclusión

El presente proyecto ha demostrado la viabilidad de los algoritmos genéticos como una herramienta eficaz para la generación de contraseñas seguras. A través de la implementación de un modelo basado en principios evolutivos, se logró optimizar la creación de claves que cumplen con criterios de seguridad como longitud mínima, diversidad de caracteres y resistencia ante ataques de fuerza bruta.

Los resultados experimentales evidenciaron que el algoritmo es capaz de mejorar progresivamente la calidad de las contraseñas generadas, alcanzando soluciones óptimas en un número relativamente bajo de iteraciones. Sin embargo, también se identificaron desafíos, como la convergencia prematura y la necesidad de diversificación en la población de soluciones, lo que sugiere la posibilidad de mejoras futuras mediante técnicas adicionales de mutación y selección.

En conclusión, este estudio demuestra que los algoritmos genéticos representan una alternativa innovadora y efectiva para la generación de contraseñas, ofreciendo un enfoque adaptable y escalable para reforzar la seguridad en entornos digitales. Su aplicación práctica, combinada con gestores de contraseñas, puede contribuir significativamente a la protección de la información y a la reducción del riesgo de accesos no autorizados.