



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLAN

AGUILAR MARTINEZ FERNANDO

TEMAS SELECTOS DE COMPUTACION

TARTEA 9  
SUDOKU 9X9 CON AG

## Tabla de contenido

1. Introducción .....	3
2. Descripción del problema.....	3
3. Espacio de estados.....	4
4. Operadores o acciones del Algoritmo Genético .....	5
1. Inicialización de la población .....	5
2. Función de Fitness .....	5
3. Selección por torneo .....	5
4. Cruza por filas .....	5
5. Mutación por intercambio .....	5
6. Elitismo .....	6
7. Actualización y convergencia .....	6
5. Estructura del proceso evolutivo en la implementación .....	6
6. Resultados.....	6
7. Conclusiones .....	7

# 1. Introducción

La resolución de sudokus es un problema clásico de búsqueda en el cual se deben completar las 81 celdas de un tablero 9×9 cumpliendo tres reglas fundamentales:

- Cada fila debe contener los dígitos del 1 al 9 sin repetirse.
- Cada columna debe contener los dígitos del 1 al 9 sin repetirse.
- Cada subcuadro 3×3 debe contener los dígitos del 1 al 9 sin repetirse.

En este proyecto se aborda la resolución de un Sudoku mediante un **Algoritmo Genético (GA)**. En lugar de utilizar búsqueda exhaustiva o backtracking, se emplea un enfoque evolutivo inspirado en procesos biológicos, donde una población de soluciones candidatas evoluciona generación tras generación gracias a operadores como selección, cruce y mutación.

Este método es especialmente interesante porque permite aproximarse a soluciones correctas incluso en problemas altamente combinatorios, utilizando heurísticas y manteniendo diversidad en la población.

El objetivo es mostrar cómo los algoritmos evolutivos pueden resolver o aproximarse a la solución de un Sudoku, así como exhibir gráficamente la evolución del fitness y la calidad de las soluciones.

## 2. Descripción del problema

### Planteamiento general

Resolver un Sudoku consiste en encontrar una asignación válida para cada una de las 81 celdas, respetando las restricciones del tablero.

Esto puede interpretarse como un problema de optimización donde cada tablero posible es un estado del sistema y el objetivo es encontrar el estado que cumple todas las reglas.

El espacio total de posibles configuraciones es enorme, por lo que métodos exactos pueden volverse muy costosos.

Los algoritmos genéticos permiten explorar este espacio mediante aproximaciones probabilísticas y mecanismos de búsqueda global.

### Representación del Sudoku

El tablero se representa como una matriz de 9×9.

En la matriz inicial, las celdas con valor 0 representan posiciones vacías.

Ejemplo:

[5 3 0 0 7 0 0 0 0]

[6 0 0 1 9 5 0 0 0]

...

Se genera una **máscara de valores fijos**, que identifica las celdas que no deben cambiarse a lo largo de la evolución.

### Entradas del sistema

El programa recibe:

- Un **tablero inicial** con dígitos y ceros para celdas vacías.
- Parámetros del algoritmo genético:
  - Tamaño de población
  - Mutación
  - Número máximo de generaciones
  - Número de élites
- Interfaz gráfica para monitorear el proceso:
  - Gráfica de fitness
  - Mejor solución
  - Tiempo transcurrido
  - Verificación automática

### Salida esperada

El sistema debe producir:

- La **mejor solución encontrada** durante la evolución.
- La gráfica del **fitness máximo por generación**.
- Validación automática:
  - Columnas correctas
  - Subcuadros correctos
  - Detección de conflictos
- El tablero completo en formato visual.

## 3. Espacio de estados

### Estado inicial

El estado inicial consiste en una población de tableros generados llenando aleatoriamente las celdas vacías fila por fila, respetando valores fijos.

Cada individuo es un tablero 9×9 válido pero no necesariamente correcto.

### Estados intermedios

Cada iteración del GA genera nuevos individuos mediante:

- Cruza de dos padres
- Mutación
- Inserción de élites

Cada estado intermedio representa una posible solución parcial del Sudoku, con distintos grados de validez.

### Estado final (meta)

La solución objetivo es un tablero con **fitness = 162**, lo que significa:

- 9 columnas con 9 valores distintos
- 9 subcuadros con 9 valores distintos

Si el GA alcanza este fitness, se considera una solución completa.

Si no se alcanza, se entrega la mejor solución disponible.

## 4. Operadores o acciones del Algoritmo Genético

### 1. Inicialización de la población

Cada individuo se genera llenando aleatoriamente los huecos de cada fila.  
Esto asegura diversidad inicial y mantiene consistencia básica.

### 2. Función de Fitness

El fitness evalúa:

- Cantidad de valores distintos en cada columna (máx. 9 por columna)
- Cantidad de valores distintos en cada subcuadro 3×3 (máx. 9 por bloque)

El fitness máximo es:

9 columnas × 9 puntos = 81

9 bloques × 9 puntos = 81

Total = 162

Mientras mayor el fitness, más consistente es la solución.

### 3. Selección por torneo

La selección elige padres mediante torneos donde tres individuos compiten y se escoge el de mayor fitness.

Esto balancea exploración y explotación.

### 4. Cruza por filas

Cada hijo se forma fila por fila:

- Se toma una fila base de alguno de los padres
- Se respetan los valores fijos
- Se corrigen duplicados
- Se rellenan los huecos faltantes sin repetir números

Este operador mantiene validez local por fila.

### 5. Mutación por intercambio

Con cierta probabilidad:

- Se seleccionan dos posiciones no fijas en una fila
- Se intercambian para generar variación

Este mecanismo evita estancarse en óptimos locales.

## 6. Elitismo

Los mejores individuos pasan directamente a la siguiente generación. Esto permite conservar las mejores soluciones y acelerar la convergencia.

## 7. Actualización y convergencia

Cada generación:

- Se computa el fitness máximo
- Se actualiza la gráfica
- Se actualiza la información de la interfaz
- Se detecta si se alcanzó la solución

## 5. Estructura del proceso evolutivo en la implementación

El flujo completo del algoritmo es:

1. Generar población inicial
2. Evaluar fitness de todos
3. Ordenar población por rendimiento
4. Guardar élités
5. Realizar selección por torneo
6. Aplicar cruce fila por fila
7. Aplicar mutación
8. Crear nueva población
9. Actualizar interfaz gráfica
10. Verificar solución
11. Repetir hasta alcanzar:
  - Fitness = 162, o
  - Límite de generaciones, o
  - Botón de “Detener” presionado

La aplicación incluye una interfaz completa que permite visualizar en tiempo real la evolución del GA y la mejor solución encontrada.

## 6. Resultados

Los resultados típicos muestran:

- Una tendencia ascendente en el gráfico de fitness
- Convergencia gradual hacia 160–162
- Comportamiento variable según mutación y tamaño de población

- Verificación automática de:
  - Columnas válidas
  - Subcuadros válidos
  - Conflictos detectados durante el proceso

En pruebas realizadas, el algoritmo puede encontrar soluciones completas o casi completas dependiendo de los parámetros de exploración.

## 7. Conclusiones

- El Sudoku puede resolverse mediante un Algoritmo Genético utilizando operadores correctamente diseñados (cruza por filas, mutación, elitismo).
- El uso de una máscara de valores fijos permite conservar la estructura original del puzzle sin romper restricciones.
- El fitness basado en columnas y bloques resulta una métrica estable y eficiente.
- La visualización en tiempo real permite comprender el comportamiento del algoritmo y su convergencia evolutiva.
- Aunque los AG no garantizan la solución óptima absoluta, con parámetros adecuados pueden resolver o aproximarse fuertemente al Sudoku.
- El proyecto demuestra que los algoritmos evolutivos son una herramienta flexible para problemas de optimización combinatoria con múltiples restricciones.