Índice

1. Definición de Algoritmo Genético (AG)

Un algoritmo genético es un método de búsqueda local que emplea una representación mediante estados para definir el problema. Es útil cuando no importa el camino al objetivo, sino solo encontrar el mejor estado según una función objetivo definida. Este algoritmo usa poca memoria y puede encontrar soluciones razonables en espacios grandes o continuos.El algoritmo comienza con un conjunto de k estados generados aleatoriamente (población) y cada estado (individuo o cromosoma), está representado (codificado) como una cadena sobre un alfabeto finito (alelos). La codificación define el tamaño del espacio de búsqueda y el tipo de operadores.

Cada elemento de un estado, individuo o cromosoma es denominado gen. El valor heurístico de un estado, individuo o cromosoma es denominado fitness, idoneidad o fenotipo y se obtiene mediante una función que mide la calidad de los estados. Este tipo de algoritmo se fundamenta en que combinando buenos estados se obtienen estados mejores. Se necesita de un conjunto de operadores (denominados operadores genéticos) que permitan generar nuevos estados.

Los algoritmos genéticos utilizan los siguientes operadores para generar nuevos estados:

• *Un operador de selección*: se encargada de escoger qué individuos van a disponer de oportunidades de reproducirse y cuáles no. Habrá individuos que aparezcan más de una vez e individuos que no aparezcan. Algunos de los más utilizados son el de ruleta y el de torneo.

• *Un operador de cruce*: su función en el algoritmo genético es que una vez seleccionados los individuos, éstos sean recombinados con una probabilidad para producir la descendencia que se insertará en la siguiente generación.

• *Un operador de mutación*: este operador provoca que algunos de los genes de un individuo varíe su valor dependiendo de la probabilidad asignada a la mutación en nuestro problema.

Algoritmo 1: Pseudo código de un algoritmo genético [1]

```
1 Inicializar individuo;
2 Calcular valor fitness;
3 repetir
4 |   Selección;
5 |   Cruce;
6 |   Mutación;
7 |   Calcular valor fitness;
8 hasta que poblacion haya convergido en solucion;
```

2. Cuestiones para el diseño e implementación

En este apartado se explicará funcionamiento clave en el desarrollo y funcionamiento del algoritmo genético implementado para la resolución de un sudoku.

2.1. Inicialización de individuo

A la hora de inicializar un individuo, se crea un array auxiliar que representa una fila cuyos valores son inicializados aleatoriamente sin repetirse. Estos valores son desde 1 hasta el tamaño del sudoku (alelos). Además, una vez inicializado este array auxiliar se comprueba que coincidan los números que hemos generado en posiciones aleatorias con los datos iniciales que se nos proporcionan ya que si no es asi, se intercambiaran los valores para que se cumpla esta restricción.

2.2. Operadores de selección

Existen dos operadores de selección en la librería de algoritmos genético utilizada en la práctica (GALib):

• *GARoulette WheelSelector*: Este tipo de operador de selección asignará una probabilidad a cada individuo según su valor fitness, esto es, cuanto más prometedor sea el individuo, mayor será la probabilidad de ser seleccionado.

• *GATournamentSelector*: Este tipo de operador establece K torneos aleatorios entre parejas de individuos y se eligen los que ganan cada torneo. El ganador de cada torneo es elegido según mejor sea su valor fitness.

2.3. Operador de cruce

El operador de cruce del sudoku comienza generando un punto de corte aleatorio dentro del tamaño del sudoku. Sin embargo, este punto debe coincidir con el principio de una fila para así copiar filas completar y seguir respetando la restricción de que no hayan elementos repetidos en filas. Para el primer descendiente se copiaran los primeros *punto1* elementos de la madre y desde *punto1* hasta el tamaño máximo del padre, simulando así un cruce por un punto en dos dimensiones. Para el segundo hijo se procederá al revés, copiando los primeros *punto1* elementos del padre y desde *punto1* hasta el final de la madre.

2.4. Operador de mutación

En cuanto a la mutación, van a existir dos formas distintas de mutar pero siempre asegurándonos de que no mutamos un dato inicial: por filas o por columnas. Una vez que nos aseguramos de que no es un dato inicial se mutará dicha casilla con una probabilidad *pmut*, cuando ya sabemos que hay que mutar se elegirá el método con una probabilidad de 0.5.

Si mutamos por columnas comprobaremos si hay repeticiones en columnas, si no hay no se mutará; en el caso de que si haya repeticiones en la columna, se coge de forma aleatoria un número que se repita (v1) y otro que no se repita (v2). Una vez hecho esto, se pondrá en la posición del numero que se repite (v1) el numero que no se repite (v2). A continuación, buscaremos la posición en la que se repite v2 en la fila y le pondremos en esa casilla el valor v1 para así asegurar que no hay repeticiones en la fila tras la mutación siempre y cuando esta posición no pertenezca a los datos iniciales porque sino hay que dejar como estaba la casilla antes para mantener esa restricción.

En cuanto a la mutación por filas, se realizará un intercambio de valores entre dos casillas que no sean datos iniciales, siguiendo respetando la restricción de que no hayan repeticiones en filas.

2.5. Condición de parada

En cuanto a la condición de parada, el algoritmo parará al encontrar que el valor fitness del individuo es 0 ya que estamos minimizando en este caso. Además, es importante que el algoritmo pare al realizar un numero máximo de generaciones ya que si el problema se hiciera muy complejo es posible que el algoritmo nunca termine ó que quizás nos interese una solución no óptima pero razonable.

Algoritmo 2: Condición de parada

```
1 si generacionActual == MAX_NUM_GENERATION or
    valorFitness ==0 entonces
2 |  parar;
3 fin
```

2.6. Función fitness

La función fitness diseñada cuenta el número de fallos del estado actual, en concreto al tratarse de un sudoku, contaremos el número de repeticiones en filas, columnas y subcuadrantes. Sin embargo, debido a que nos aseguramos en todo momento de que el individuo no tenga repeticiones en las filas, solo será necesario contar las repeticiones de números en columnas y subcuadrantes. Por tanto, cuanto menor sea el valor fitness mejor ya que significará que hay menos fallos en el sudoku, significando un valor fitness 0 que se ha resuelto el sudoku satisfactoriamente.

3. Casos para el ajuste

A continuación, se muestran la tabla que recoge los datos obtenidos tras la ejecución del algoritmo para los casos de ajuste junto con el promedio de valor fitness para indicar que configuraciones encuentran la solución óptima en todos los caso de ajuste.

Para representar las soluciones óptimas se ha marcado en verde las celdas en las cuales el promedio valor fitness es 0 para todos los casos de ajuste. Sin embargo, se han marcado en amarillo en los casos que se obtiene la solución óptima para un caso dado. Además, se adjunta la tabla de datos completas en un fichero externo para mejor visualización.

				A1		A2		A3		A4		A5		A6		
Selector	Tamaño de poblacion	Prob.de Cruce	Prob.de mutacion	Fitness	Generacione	Fitness	Generacione	Fitness	Generaciones	Fitness	Generaciones	Fitness	Generaciones	Fitness	Generaciones	
GARouletteWheelSelector	100	0.8	0.05	2	12000	4	12000	4	12000	2	12000	2	12000	2	12000	2,66666667
GARouletteWheelSelector	100	0.8	0.075	2	12000	0	1046	0	9534	2	12000	0	3623	4	12000	1,33333333
GARouletteWheelSelector	100	0.8	0.1	2	12000	4	12000	0	11485	4	12000	2	12000	2	12000	2,33333333
GARouletteWheelSelector	100	0.8	0.125	2	12000	0	4065	0	8889	5	12000	2	12000	2	12000	1,83333333
GARouletteWheelSelector	100	0.85	0.05	2	12000	0	11072	0	11894	0	1840	0	337	0	6651	0,33333333
GARouletteWheelSelector	100	0.85	0.075	0	1078	0	742	4	12000	4	12000	4	12000	7	12000	3,16666667
GARouletteWheelSelector	100	0.85	0.1	0	218	0	4331	2	12000	2	12000	0	6007	0	6415	0,66666667
GARouletteWheelSelector	100	0.85	0.125	2	12000	4	12000	6	12000	4	12000	4	12000	2	12000	3,66666667
GARouletteWheelSelector	100	0.9	0.05	2	12000	0	763	2	12000	0	3013	2	12000	2	12000	1,33333333
GARouletteWheelSelector	100	0.9	0.075	0	5178	2	12000	0	11871	4	12000	0	5066	2	12000	1,33333333
GARouletteWheelSelector	100	0.9	0.1	2	12000	4	12000	0	6630	2	12000	4	12000	6	12000	3
GARouletteWheelSelector	100	0.9	0.125	2	12000	0	9920	0	2481	7	12000	4	12000	4	12000	2,83333333
GARouletteWheelSelector	100	0.95	0.05	2	12000	0	1304	4	12000	2	12000	0	4325	2	12000	1,66666667
GARouletteWheelSelector	100	0.95	0.075	0	8375	4	12000	2	12000	4	12000	5	12000	0	4381	2,5
GARouletteWheelSelector	100	0.95	0.1	0	3498	7	12000	2	12000	4	12000	4	12000	0	1416	2,83333333
GARouletteWheelSelector	100	0.95	0.125	0	4495	0	6570	0	2793	0	2619	0	2270	2	12000	0,33333333
GARouletteWheelSelector	150	0.8	0.05	0	10221	4	12000	0	1522	2	12000	0	3680	6	12000	2
GARouletteWheelSelector	150	0.8	0.075	3	12000	4	12000	2	12000	2	12000	4	12000	0	162	2,5
GARouletteWheelSelector	150	0.8	0.1	2	12000	0	2115	0	6343	2	12000	4	12000	2	12000	1,66666667
GARouletteWheelSelector	150	0.8	0.125	2	12000	0	6885	4	12000	2	12000	6	12000	2	12000	2,66666667
GARouletteWheelSelector	150	0.85	0.05	0	11411	0	1271	2	12000	4	12000	0	5223	0	531	1
GARouletteWheelSelector	150	0.85	0.075	0	8494	2	12000	4	12000	4	12000	5	12000	2	12000	2,83333333
GARouletteWheelSelector	150	0.85	0.1	0	3301	2	12000	2	12000	0	4272	6	12000	4	12000	2,33333333
GARouletteWheelSelector	150	0.85	0.125	2	12000	8	12000	2	12000	4	12000	4	12000	4	12000	4
GARouletteWheelSelector	150	0.9	0.05	0	8243	0	412	5	12000	0	2342	2	12000	6	12000	2,16666667
GARouletteWheelSelector	150	0.9	0.075	4	12000	4	12000	0	2924	4	12000	0	855	2	12000	2,33333333
GARouletteWheelSelector	150	0.9	0.1	0	6418	4	12000	2	12000	0	8201	7	12000	7	12000	3,33333333
GARouletteWheelSelector	150	0.9	0.125	8	12000	6	12000	2	12000	7	12000	6	12000	0	9381	4,83333333
GARouletteWheelSelector	150	0.95	0.05	2	12000	0	8400	0	4104	4	12000	0	6652	4	12000	1,66666667
GARouletteWheelSelector	150	0.95	0.075	4	12000	0	548	0	2002	4	12000	2	12000	2	12000	2
GARouletteWheelSelector	150	0.95	0.1	4	12000	0	1408	0	1509	3	12000	2	12000	2	12000	1,83333333
GARouletteWheelSelector	150	0.95	0.125	0	4339	0	390	2	12000	2	12000	4	12000	0	7225	1,33333333
GATournamentSelector	100	0.8	0.05	3	12000	0	104	0	4913	2	12000	0	68	0	10638	0,83333333
GATournamentSelector	100	0.8	0.075	0	72	0	4107	0	3457	0	563	0	80	0	72	0
GATournamentSelector	100	0.8	0.1	0	80	0	6216	0	117	0	65	4	12000	0	7314	0,66666667
GATournamentSelector	100	0.8	0.125	0	10391	0	8658	0	5343	0	908	0	133	0	495	0
GATournamentSelector	100	0.85	0.05	2	12000	0	34	4	12000	2	12000	0	845	2	12000	1,66666667
GATournamentSelector	100	0.85	0.075	0	4824	0	347	0	1873	0	297	0	4757	0	1605	0
GATournamentSelector	100	0.85	0.1	0	373	0	6799	0	2989	0	2392	2	12000	0	4255	0,33333333
GATournamentSelector	100	0.85	0.125	2	12000	0	123	0	1085	0	5941	0	567	0	4948	0,33333333
GATournamentSelector	100	0.9	0.05	2	12000	0	368	0	127	0	146	0	53	4	12000	1
GATournamentSelector	100	0.9	0.075	0	1991	0	2870	0	447	2	12000	0	6614	0	334	0,33333333
GATournamentSelector	100	0.9	0.1	0	52	0	47	0	811	2	12000	0	10319	0	131	0,33333333
GATournamentSelector	100	0.9	0.125	0	11507	0	82	0	445	2	12000	2	12000	0	6236	0,66666667
GATournamentSelector	100	0.95	0.05	0	1014	0	825	0	40	0	4760	0	5977	2	12000	0,33333333
GATournamentSelector	100	0.95	0.075	0	1289	0	25	2	12000	0	406	0	71	0	611	0,33333333
GATournamentSelector	100	0.95	0.1	2	12000	0	91	2	12000	0	1798	0	6666	0	863	0,66666667
GATournamentSelector	100	0.95	0.125	0	6590	0	109	0	174	2	12000	2	12000	0	473	0,66666667
GATournamentSelector	150	0.8	0.05	2	12000	0	56	4	12000	0	32	0	57	0	1182	1
GATournamentSelector	150	0.8	0.075	0	50	0	53	0	577	0	84	0	8040	4	12000	0,66666667
GATournamentSelector	150	0.8	0.1	0	1062	0	931	0	115	2	12000	0	4019	0	755	0,33333333
GATournamentSelector	150	0.8	0.125	0	384	0	1036	0	537	0	2549	2	12000	0	625	0,33333333
GATournamentSelector	150	0.85	0.05	2	12000	0	44	2	12000	4	12000	2	12000	0	458	1,66666667
GATournamentSelector	150	0.85	0.075	2	12000	0	78	0	661	2	12000	0	146	0	5911	0,66666667
GATournamentSelector	150	0.85	0.1	0	673	0	135	0	121	0	924	0	2630	0	306	0
GATournamentSelector	150	0.85	0.125	0	4690	0	114	2	12000	0	8515	0	9452	2	12000	0,66666667
GATournamentSelector	150	0.9	0.05	0	104	4	12000	0	53	0	111	0	49	0	1029	0,66666667
GATournamentSelector	150	0.9	0.075	0	146	0	280	0	119	0	578	0	58	0	239	0
GATournamentSelector	150	0.9	0.1	0	67	0	266	0	183	0	2881	0	380	0	486	0
GATournamentSelector	150	0.9	0.125	0	2670	0	76	0	830	0	1887	0	5002	0	5489	0
GATournamentSelector	150	0.95	0.05	2	12000	0	860	0	11813	2	12000	2	12000	0	36	1
GATournamentSelector	150	0.95	0.075	0	49	0	144	0	288	0	275	0	76	0	2717	0
GATournamentSelector	150	0.95	0.1	2	12000	0	47	0	84	2	12000	0	60	0	9172	0,66666667
GATournamentSelector	150	0.95	0.125	0	5075	0	2081	0	264	0	362	0	108	0	1415	0

Figura 1: Casos para el ajuste

4. Análisis de las pruebas experimentales sobre los casos de ajuste

En primer lugar, podemos ver que el comportamiento del método de selección GATournamentSelector es mucho mejor que el de GARouletteSelector ya que este resuelve el sudoku (fitness 0) para todos los casos de ajuste en numerosas configuraciones mientras que el de ruleta no resuelve con una misma configuración todos los casos.

En cuanto al tamaño de la población podemos ver que el ligeramente mejor tener un tamaño de población de 150 frente a uno de 100 ya que es más consistente a la hora de conseguir soluciones óptimas sin importar el caso de ajuste.

Sin embargo, podemos observar que la probabilidad de mutación suele comportarse bastante bien en tanto 0.075 como en 0.125, pero quizás la probabilidad de mutación que mejor se comporta es 0.075. En cuanto a la probabilidad de cruce se puede ver en la tabla que al juntar una probabilidad de cruce de 0.9 al juntarlo con un tamaño de población de 150 y el selector de torneo con casi cualquier probabilidad de mutación da lugar a buenos resultados. Por tanto, elegiremos normalmente una probabilidad de cruce de 0.9.

5. Manual de asignación

Por tanto, a la hora de recomendar una configuración por tanto velocidad ya que converge en la solución óptima antes y por encontrar dicha solución óptima es:

- Selector: GATournamentSelector
- Tamaño de población: 150
- Probabilidad de cruce: 0.9
- Probabilidad de mutacion: 0.075

Para asignar estos parámetros debemos seguir el siguiente procedimiento:
-Ejecutar en línea de comandos: Sudoku.exe -t 150 0.9 0.075 archivo.txt

6. Casos del usuario

A continuación, se muestran los casos de usuario con la configuración recomendada en el manual de asignación:

Caso: Usuario-1.txt
Entrada: >Sudoku.exe -t 150 0.9 0.075 Usuario-1.txt

Salida:

```
Selector: GATournamentSelector
Tam de poblacion: 150
Probabilidad de cruce: 0.9
Probabilidad de mutacion: 0.075
Archivo: Usuario-1.txt
Fitness: 0
Numero de generaciones: 54
Estado: RESUELTO

Estado final del sudoku:

[ 9 2 7 1 5 3 4 6 8 ]
[ 4 5 3 2 6 8 7 9 1 ]
[ 1 6 8 9 7 4 3 5 2 ]
[ 2 3 9 4 1 7 5 8 6 ]
[ 5 8 1 3 9 6 2 4 7 ]
[ 7 4 6 8 2 5 1 3 9 ]
```

```
[ 8 9 5 7 4 2 6 1 3 ]
[ 3 7 4 6 8 1 9 2 5 ]
[ 6 1 2 5 3 9 8 7 4 ]
```

Caso: Usuario-2.txt
Entrada:>Sudoku.exe -t 150 0.9 0.075 Usuario-2.txt

Salida:

```
Selector: GATournamentSelector
    Tam de poblacion: 150
    Probabilidad de cruce: 0.9
    Probabilidad de mutacion: 0.075
    Archivo: Usuario-2.txt
    Fitness: 0
    Numero de generaciones: 35
    Estado: RESUELTO

Estado final del sudoku:

[ 5 1 9 7 4 6 2 8 3 ]
[ 3 8 4 5 2 9 7 6 1 ]
[ 6 2 7 8 1 3 5 4 9 ]
[ 7 3 1 4 6 5 8 9 2 ]
[ 4 5 8 3 9 2 6 1 7 ]
[ 9 6 2 1 8 7 4 3 5 ]
[ 8 9 5 2 3 4 1 7 6 ]
[ 1 7 6 9 5 8 3 2 4 ]
[ 2 4 3 6 7 1 9 5 8 ]
```

Caso: Usuario-3.txt
Entrada:>Sudoku.exe -t 150 0.9 0.075 Usuario-3.txt

Salida:

```
Selector: GATournamentSelector
    Tam de poblacion: 150
    Probabilidad de cruce: 0.9
    Probabilidad de mutacion: 0.075
    Archivo: Usuario-3.txt
    Fitness: 0
    Numero de generaciones: 54
    Estado: RESUELTO

Estado final del sudoku:

[ 9 2 7 1 5 3 4 6 8 ]
[ 4 5 3 2 6 8 7 9 1 ]
[ 1 6 8 9 7 4 3 5 2 ]
[ 2 3 9 4 1 7 5 8 6 ]
[ 5 8 1 3 9 6 2 4 7 ]
[ 7 4 6 8 2 5 1 3 9 ]
```

```
[ 8 9 5 7 4 2 6 1 3 ]
[ 3 7 4 6 8 1 9 2 5 ]
[ 6 1 2 5 3 9 8 7 4 ]
```

Caso: Usuario-4.txt
Entrada:>Sudoku.exe -t 150 0.9 0.075 Usuario-4.txt

Salida:

```
    Selector: GATournamentSelector
        Tam de poblacion: 150
        Probabilidad de cruce: 0.9
        Probabilidad de mutacion: 0.075
        Archivo: Usuario-4.txt
        Fitness: 0
        Numero de generaciones: 0
        Estado: RESUELTO

Estado final del sudoku:

        [ 1 2 4 3 ]
        [ 3 4 2 1 ]
        [ 2 1 3 4 ]
        [ 4 3 1 2 ]
```

Referencias

[1] Vijini Mallawaarachchi ,towards data science, *Introduction to Genetic Algorithms*, Pseudo código de un algoritmo genético genérico.
Available: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3

[2] Mata Pérez, Miguel, *Bibliografía en LATEX*, Utilizado para el desarrollo de referencias en LATEX.
Available: http://logistica.fime.uanl.mx/miguel/docs/BibTeX.pdf.

[3] Profesores de la asignatura de sistemas inteligentes de la universidad de murcia, *Apuntes de la asignatura de sistemas inteligentes*, Utilizado para el desarrollo del algoritmo genético.