



Universidad de Carabobo
Facultad Experimental de Ciencias y Tecnología
Departamento de Computación
Electiva: Introducción a la Inteligencia Artificial



Tarea 1: Algoritmos Genéticos simples

Integrantes:

Jhonattan Garcia, C.I: 24.423.188

Luis Sierra, C.I: 25.582.661

Prof. Aníbal Guerra

Bárbula, 07 de septiembre de 2021

Pasos para ejecutar el programa

- 1- Tener instalado el intérprete de Python en su S.O.
- 2- Crear un entorno virtual con el siguiente comando:
 - Windows: `py -m venv venv`
 - Linux: `python3 venv venv`
- 3- Ejecutar el programa
 - Windows: `py main.py`
 - Linux: `python3 main.py`

Datos utilizados para el programa

Los datos de prueba se encuentran en el archivo **in.txt** donde cada línea corresponde a un caso de prueba distinto. Cada línea contiene 5 elementos separados por un espacio en blanco que corresponden a: N, Pc, Pm, #iter, Gap.

Entrada:

N	PC	PM	#iter	Gap
10	0.1	0.001	100	10
100	0.8	0.5	1000	100
10	0.1	0.5	100	100
10	0.8	0.001	100	100
10	0.8	0.001	100	10
100	0.1	0.5	1000	100
100	0.1	0.5	1000	10
100	0.8	0.001	1000	100
100	0.8	0.001	1000	10

Salida: para cada caso de prueba se generará un archivo outN.txt con N = 1, 2..., n correspondiente al reporte de las generaciones.

Observaciones

Generales

1. El programa puede tardar en generar el archivo de salida ya que al realizar un proceso de cruce o mutación se pueden crear individuos que no pertenezcan al espacio de soluciones y de ser así, se repetirá dicho proceso hasta que aparezcan los individuos permitidos dentro del rango [0, 1].
2. No todas las veces que se ejecuten los mismos casos de prueba se obtienen los resultados esperados.

Casos de prueba adicionales

1.

N	PC	PM	#iter	Gap
10	0.1	0.006	100	60

Al realizar el experimento se comprobó que utilizando valores mayores a 60% para el Gap y un número mayor a 100 generaciones, el tiempo de respuesta del programa es mucho más lento debido a la cantidad de individuos nuevos que van apareciendo a través de las nuevas generaciones. Adicionalmente, se verificó con una cantidad menor a 100 generaciones, pero se observó que el resultado no es el esperado.

Las imágenes que se presentan a continuación permiten visualizar la convergencia de la función con los datos introducidos.



```
1 GENERACION ACTUAL: 0
2 Mejor adaptacion: 1.8238739, Adaptacion promedio: 1.1329022
3
4 No  Cromosoma                Valor real  Adaptacion
5 1   0111111011000101001011   0.8308043  1.8238739
6 2   01111100110010000100001  0.8177697  1.6780857
7 3   000011011101001110101100  0.906156   1.5675367
8 4   011101000110111010111101  0.7630525  1.4187161
9 5   010000100111110100010100  0.4357396  1.4170769
10 6   001001110101111000000110  0.2579974  1.1783031
11 7   001000000101000101111110  0.2118014  0.6863574
12 8   00101011010111111101111  0.2842607  0.6068233
13 9   000000001001101101001100  0.39756    0.5524829
14 10  001011000110101111111001  0.2911225  0.3997656
15
```

1.a. Población inicial de 10 individuos aleatorios en [0, 1].

```
1 GENERACION ACTUAL: 100
2 Mejor adaptacion: 1.9244451, Adaptacion promedio: 0.7740596
3
4 No  Cromosoma          Valor real  Adaptacion
5 1  100011110001011110000101  0.9377669  1.9244451
6 2  000011100101000000011101  0.938013   1.9233794
7 3  100011011001001011000101  0.9278149  1.9158182
8 4  10001111011010001111100   0.9417852  1.8994564
9 5  000000000001010010001111  0.0005263  0.0173671
10 6  000000000001010001010111  0.0005207  0.0171823
11 7  000000000001010001010111  0.0005207  0.0171823
12 8  000000000001010001010101  0.0005205  0.0171757
13 9  000000000000010100010110  0.0001302  0.0042966
14 10 000000000000010100010101  0.0001301  0.0042933
15
```

1.b. Población descendiente después de 100 generaciones.

2.

N	PC	PM	#iter	Gap
100	0.7	0.002	100	20

Con este experimento observamos que a medida que se incrementa la cantidad de individuos, si la cantidad de generaciones es mayor a 100 y el Gap es mayor al 20%, el tiempo de respuesta es lento.

En las imágenes que se muestran a continuación se puede apreciar la convergencia de la función con estos parámetros.



```
1  GENERACION ACTUAL: 0
2  Mejor adaptacion: 1.9258245, Adaptacion promedio: 1.1394185
3
4  No  Cromosoma          Valor real    Adaptacion
5  1   000011100011000000111110  0.929854     1.9258245
6  2   100011011010101001001111  0.9284175    1.9192149
7  3   100011111000101011001110  0.9407182    1.9076581
8  4   011111110001000010001001  0.8327305    1.8311539
9  5   10001011110010000000101  0.9160709    1.7784422
10 6   100000010011110000110011  0.8469555    1.7683905
11 7   100000010110111111010110  0.8482774    1.7524578
12 8   100010101110001010010101  0.9101973    1.6627828
13 9   011011101001100111101010  0.7248362    1.6581773
14 10  100000101001100010100101  0.8558757    1.6306206
15 11  011000001101011110011100  0.6346652    1.6285037
16 ...
17 92  0001111111110010100101001  0.2090281    0.6037029
18 93  000011010100010000000101  0.0869381    0.4388131
19 94  000011010111000010001001  0.0880777    0.4055913
20 95  000001000111010101000101  0.292165     0.3675913
21 96  000011111101110101101101  0.1039725    0.2884374
22 97  000011101000100001001011  0.0952395    0.1890301
23 98  000011101011101011011101  0.0965341    0.1490114
24 99  000011101110011110010000  0.0976784    0.1135616
25 100 000011101110111100000100  0.0978692    0.1076473
26
```

2.a. Población inicial de 100 individuos aleatorios en $[0, 1]$.



```
1  GENERACION ACTUAL: 100
2  Mejor adaptacion: 1.9258245, Adaptacion promedio: 1.6186474
3
4  No  Cromosoma          Valor real    Adaptacion
5  1   000011100011000000111110  0.929854     1.9258245
6  2   100011011010101001001111  0.9284175    1.9192149
7  3   100011111000101011001110  0.9407182    1.9076581
8  4   011111110001000010001001  0.8327305    1.8311539
9  5   000011001101101011011000  0.842456     1.8101056
10 6   000011001110011101011001  0.845657     1.7824363
11 7   100010111100100000000101  0.9160709    1.7784422
12 8   100000010011110000110011  0.8469555    1.7683905
13 9   100000010110111111010110  0.8482774    1.7524578
14 10  011100000101010010101000  0.7361704    1.7361603
15 11  011100001011110101010111  0.7388503    1.7355501
16 ...
17 92  011001100101001110110001  0.6706097    1.1775558
18 93  000010100011110000000111  0.670727     1.174434
19 94  001101111000101011000101  0.3640005    1.1586042
20 95  001110001101110000110101  0.3726389    0.9713779
21 96  000001011010111110110010  0.372658     0.9709074
22 97  011001111110110011000011  0.6810819    0.876347
23 98  011001111110111111101000  0.6811624    0.8739005
24 99  010011001010010101100100  0.5023076    0.8600787
25 100 001011000110011101010010  0.2910034    0.4034343
26
```

2.b. Población descendiente después de 100 generaciones.

Listado del programa

Función	Entrada	Salida
<i>float_to_bin</i>	x: número real, n_bits: cantidad de bits permitidos	Retorna el equivalente en binario de x
<i>create_population</i>	n: cantidad de individuos, n_bits: cantidad de bits permitidos	Genera una población de N individuos únicos.
<i>evaluate</i>	population: lista de individuos	Agrega el valor de la adaptación a cada individuo de la población.
<i>add_adaptations</i>	population: lista de individuos	Retorna la suma de todas las adaptaciones de los individuos que forman la población de entrada
<i>show_progress</i>	population: lista de individuos, cases: número de caso de prueba actual, generations: generación actual	Muestra la estadística de la generación actual

<i>selection_probability</i>	population: lista de individuos	Agrega el valor de la probabilidad de selección a cada individuo de la población
<i>search</i>	search_space: espacio de búsqueda que corresponde a la lista de la rueda de la ruleta, elem: elemento a comparar con los valores del espacio de búsqueda	Retorna el índice del individuo donde se encuentre elem
<i>roulette_wheel</i>	population: lista de individuos	Retorna dos individuos aleatorios
<i>encode</i>	num: número real, n_bits: umero de bits permitidos	Dado un número real, obtiene su correspondiente número entero
<i>decoding</i>	num: número entero, n_bits: umero de bits permitidos	Dado un número entero, obtiene su correspondiente número real

<i>scientific_notation</i>	num: número real	Convierte el número real de formato x.xxe-0x a 0.0000xxx
<i>modification_point</i>	prob: probabilidad de cruce o mutación, n_bits: cantidad de bits permitidos	Retorna el punto de modificación de bits para el cruce y la mutación
<i>crossing</i>	parent1 y parent2: números reales que representan a los padres, pc: probabilidad de cruce, n_bits: cantidad de bits permitidos	Retorna 2 hijos de dos individuos
<i>mutation</i>	p1 y p2: números reales que representan a los individuos, pc: probabilidad de cruce, n_bits: cantidad de bits permitidos	Retorna la mutación de p1 y p2
<i>main</i>		Cuerpo principal del AGS