Instituto Politécnico Nacional

Escuela Superior de Computo

Algoritmos Genéticos

Práctica 2: Individuos

Hernández López Ángel Zait

2014080682

3CM5

Periodo: 2019/01

Planteamiento del problema
Realizar un programa con 4 tipos de representación de individuos. Deben llenar un arreglo o matiz con 10 individuos, cada alelo se creará de manera aleatoria.

Introducción
Primero debemos saber que es un cromosoma, gene, individuo y un alelo.
El cromosoma , en algoritmos genéticos es una estructura de datos que contienen una cadena de parámetros de diseño o genes. El gene, es una subsección de un cromosoma que codifica el valor de un solo parámetro. El individuo es un solo miembro de la población de soluciones potenciales a un problema. Cada individuo contiene un cromosoma (o de manera más general, un genoma) que representa una solución posible al problema a resolverse. Y el alelo es cada valor posible que puede adquirir una cierta posición genética.
Hay distintas formas de representar a un individuo, los cuales son:

- Binaria: Es la representación tradicional usada para codificar un conjunto de soluciones es el esquema binario en el cual un cromosoma1 es una cadena de la forma (b1, b2, . . . , bm), donde b1, b2, . . . , bm se denominan alelos (ya sea ceros o unos). Hay varias razones por las cuales suele usarse la codificación binaria en los Ags, aunque la mayoría de ellas se remontan al trabajo pionero de Holland en el área.
- Código Gray: la representación binaria no mapea adecuadamente el espacio de búsqueda con el espacio de representación. La codificación de Gray es parte de una familia de representaciones Podemos convertir cualquier n´ umero binario a un código de Gray haciendo XOR a sus bits consecutivos de derecha a izquierda. Por ejemplo, dado el número 101 en binario, haríamos5: $1 \oplus 0 = 1$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, produciéndose (el último bit de la izquierda permanece igual) 0111, el cual es el código de Gray equivalente. Algunos investigadores han demostrado empíricamente que el uso e códigos de Gray mejora el desempeño del AG.
- Números reales
  - Flotante: Si queremos codificar una variable que va de 0.35 a 1.40 usando una precisi´ on de 2 decimales, necesitaríamos $\log_2 (140 - 35) \approx 7$ bits para representar cualquier n´ umero real dentro de ese rango. Sin embargo, en este caso, tenemos el mismo problema del que hablamos anteriormente, porque el número 0.38 se representar´ la como 0000011, mientras que 0.39 se representar´ la como 0000101.
  - Enteros: Una representación entera de números reales. La cadena completa es decodificada como un solo número real multiplicando y dividiendo cada dígito de acuerdo a su posición.

Contenido
Primero se creo una pequeña lista, para saber que tipo de individuo quiere el usuario, los cuales son binario, codigo Gray, real flotante y real entero. A su vez se hizo una pequeña librería donde se encuentran los procesos para crear los individuos con alelos aleatorios. Veamos las funciones que nos ayudaron a la creación de individuos que se encuentran en la librería “individuos.h”. Hay que adelantar que se generaron 10 individuos con 10 alelos cada uno.

- void indiBool(void):

Genera individuos de tipo binario; se asignó un matriz donde se guardarían los alelos. Para generar la información se utilizó un generador de números aleatorios que fuera del 0 al 1, ya que son los únicos valores que en código binario, se guardan y se manda a imprimir.

```
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$ g++ practica2.cpp -o pra
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$ ./prac2
Generar individuo de tipo
1. Binario
2. Código Gray
3. Reales
4. Enteros
Opcion: 1

              Individuo Binario
Individuo  1: 1 1 1 0 1 1 1 0 0 0
Individuo  2: 1 1 0 1 1 0 0 1 0 0
Individuo  3: 0 0 0 1 0 1 1 0 1 0
Individuo  4: 0 0 1 0 0 0 1 0 0 1
Individuo  5: 0 1 0 0 0 1 1 0 1 1
Individuo  6: 0 1 1 1 1 1 0 0 1 1
Individuo  7: 0 0 0 1 0 0 1 1 0 1
Individuo  8: 1 0 0 1 1 0 1 0 0 0
Individuo  9: 1 1 1 1 0 0 0 0 1 0
Individuo 10: 0 1 0 0 1 0 0 0 1 1
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$
```

- Void indiGray(void):

Genera individuos de tipo código Gray, en este caso, se generó primero una población binaria, utilizando la misma estructura que en la función anterior, la diferencia es que se aplica la operación 'XOR' en cada uno de los bits de derecha a izquierda, dejando el ultimo bit intacto, ya que será su mismo valor. Para notar la diferencia, se imprimen ambos individuos, el tipo binario, y el tipo código Gray.

```
              Individuo Codigo Gray
Individuo   1: 0 1 0 1 1 1 0 1 0 0
Individuo   2: 1 1 0 1 0 1 0 1 1 1
Individuo   3: 1 0 0 0 0 0 1 1 0 1
Individuo   4: 0 1 0 0 0 1 1 0 1 0
Individuo   5: 0 0 1 1 0 0 0 0 1 0
Individuo   6: 1 0 0 1 1 0 0 0 1 0
Individuo   7: 1 0 1 1 0 0 1 0 0 0
Individuo   8: 0 1 0 1 0 0 1 0 0 1
Individuo   9: 0 0 1 0 1 0 0 1 1 0
Individuo  10: 0 0 0 1 0 0 1 1 0 0

Individuo  1': 0 1 1 1 0 0 1 1 1 0
Individuo  2': 1 0 1 1 1 1 1 1 0 0
Individuo  3': 1 1 0 0 0 0 0 1 0 1 1
Individuo  4': 0 1 1 0 0 1 0 1 1 1
Individuo  5': 0 0 1 0 1 0 0 0 1 1
Individuo  6': 1 1 0 1 0 1 0 0 1 1
Individuo  7': 1 1 1 0 1 0 1 1 0 0
Individuo  8': 0 1 1 1 1 0 1 1 0 1
Individuo  9': 0 0 1 1 1 1 1 0 1 0 1
Individuo 10': 0 0 0 1 1 0 1 0 1 0
```

- Void iniReal(void):

Genera individuos de tipo entero decimal o flotante; se usó el mismo método que en los anteriores, generar números aleatorios, pero ahora el rango va de 0 a 999, esto para después almacenar ese numero a una variable auxiliar, y enseguida dividirla entre 100, para que de un valor decimal, y a continuación, guardarla en la matriz. Después se imprimen los individuos.

```
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$ ./prac2
Generar individuo de tipo
1. Binario
2. Código Gray
3. Reales
4. Enteros
Opcion: 3

                Individuo Real Flotante
Individuo  1:  5.93  0.07  1.16  6.76  5.32  2.05  4.07  0.61  4.45  5.69
Individuo  2:  7.69  1.73  5.47  0.44  5.67  9.51  6.90  2.11  6.73  3.38
Individuo  3:  8.68  7.03  7.47  0.19  2.03  9.91  9.41  6.66  1.41  7.42
Individuo  4:  1.30  7.34  7.49  2.47  7.63  2.81  4.52  1.70  6.94  2.49
Individuo  5:  0.91  4.63  7.74  9.90  8.59  3.42  9.41  9.01  9.05  9.66
Individuo  6:  5.92  1.25  6.69  3.39  4.96  2.25  3.31  4.38  8.91  8.24
Individuo  7:  1.80  0.21  9.11  2.82  6.20  6.74  9.15  4.24  8.44  6.10
Individuo  8:  0.26  2.87  4.25  8.00  2.77  6.37  4.94  2.18  5.38  7.51
Individuo  9:  1.84  1.30  8.76  2.05  4.70  3.73  4.30  8.01  1.63  6.73
Individuo 10:  9.77  6.95  6.95  8.88  9.77  6.67  5.62  2.45  4.44  7.58
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$
```

- Void indiEnte(void):

Genera individuos de tipo entero; como en indiBool(), se generarán números aleatorios, en este caso son números que van de 0 al 99, luego de haber generado el número aleatorio, se guarda en la matriz, y luego se manda a imprimir.

```
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$ g++ practica2.cpp -o pra
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$ ./prac2
Generar individuo de tipo
1. Binario
2. Código Gray
3. Reales
4. Enteros
Opcion: 4

               Individuo Real Entero
Individuo  1:   65   97   27   68   71   45   82   47    9   61
Individuo  2:   63   70   64   30    6   19   47    9   62   38
Individuo  3:    2   82   53   22   32   61   78   16    4   25
Individuo  4:   67   21   23   94   89   94   40   72   93    1
Individuo  5:   85    9   72    1   91   30   21   90   39   35
Individuo  6:   80   41   70   33   63   54   47   94   71   51
Individuo  7:   19   38   24   94   33   13   40   25   37   86
Individuo  8:   26   22   95   50   76   86   80   49   28   19
Individuo  9:   84    8   13    6   41   76   61   88   22   32
Individuo 10:   91   42   22   67   36    7   81   29   32   18
zait@zait-Satellite-L845:~/Documentos/AG/Practica2$
```

Conclusiones

Se pudo observar los distintos tipos de representaciones de un individuo, esto porque no en todas los problemas usaran bits, u otro tipo. Ya que puede llegar el caso de que cada unon de los individuos sea más fácil, a futuro, realizar alguna de las operaciones de los algoritmos genéticos.