INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Genetic Algorithms

Practica 2:

Profa.: MORALES GUITRON SANDRA LUZ

Grupo: 3CM5

Alumno:

Salcedo Barrón Ruben Osmair.

MEXICO, D.F. a 13 de septiembre del 2018

Introducción

Se denomina individuo a 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.

Representación Binaria. La representación usada por el algoritmo genético. 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, bum), donde b1, b2, bum se denominan alelos (ya sea ceros o unos). Hay varias razones por las cuales suele usarse la codificación binaria en los Agus, aunque la mayoría de ellas se remontan al trabajo pionero de Hollande en el área.

Códigos de Gray. La investigación en AGs fue que el uso de 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´uñero binario a un código de Gray haciendo XOR a sus bits consecutivos de derecha a izquierda. Por ejemplo, dado el número101 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 códigos de Gray mejora el desempeño del AG.

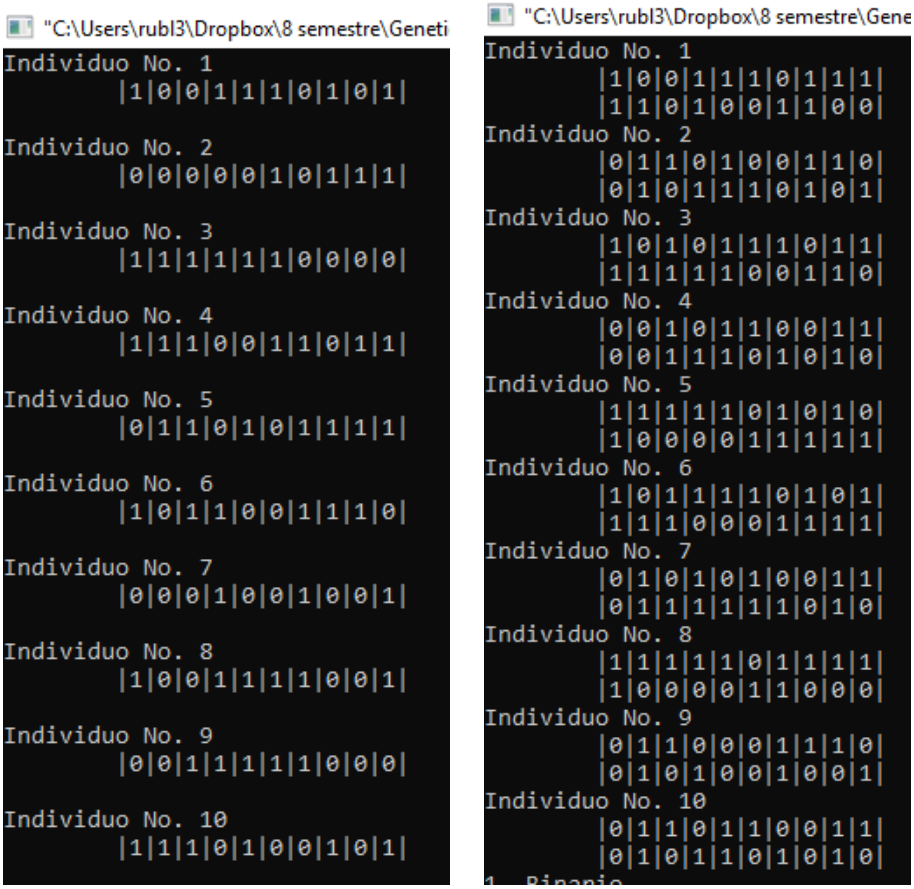
Codificación en números reales. El espacio de búsqueda en el espacio de representación se vuelve más serio cuando tratamos de codificar números reales. Por ejemplo, si queremos codificar una variable que va de 0.35a 1.40 usando una precisión de 2 decimales, necesitaríamos $\log2(140 - 35) \approx 7$ bits para representar cualquier número real dentro de ese rango.

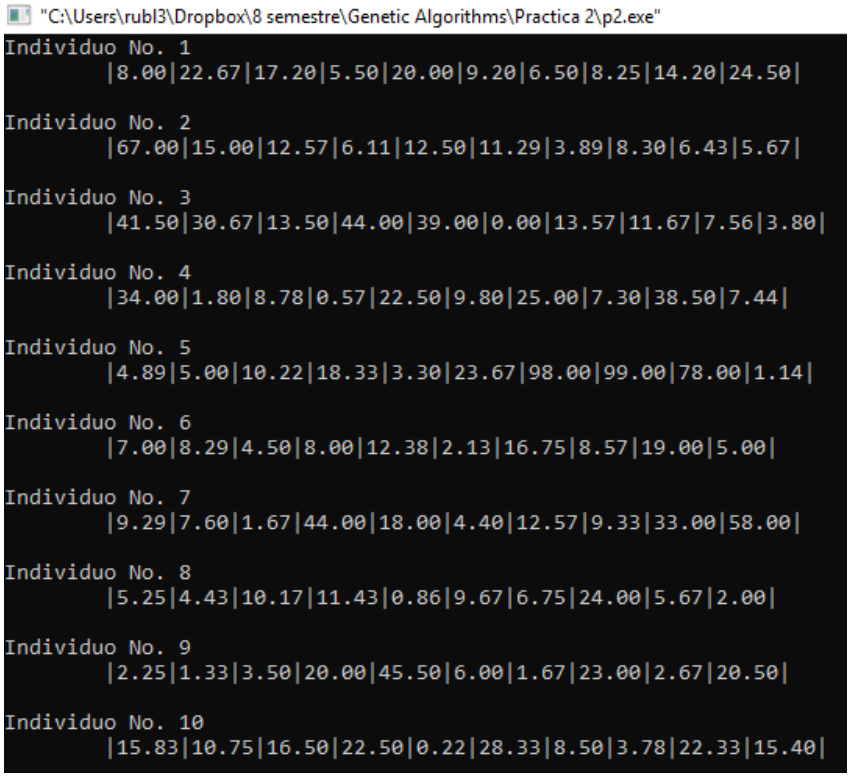
Codificación en números enteros. Una representación entera de números reales. La cadena completa es decodificada como un solo número real multiplicando y dividiendo cada dígito de acuerdo a su posición

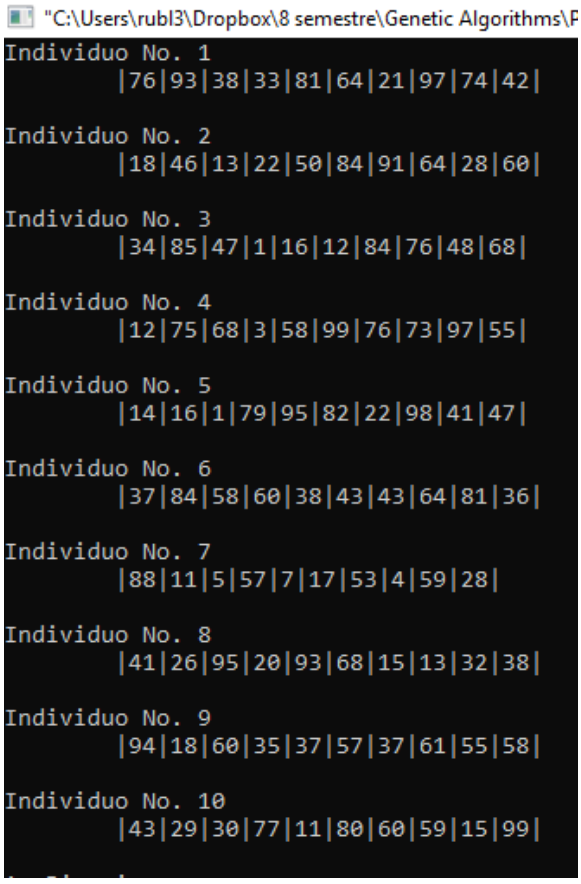
Desarrollo

En esta práctica el desarrollo igual que la práctica fue sencilla solo que al momento de leer las instrucciones hubo confusiones y se realizó como mejor se interpretó. A pesar de eso el código genera los individuos mediante arreglos muy

sencillos los cuales dependiendo que se escoja se llenan aleatoriamente esto representando a 10 individuos.




"C:\Users\rubl3\Dropbox\8 semestre\Genetic Algorithms\Practica 2\p2.exe"
Individuo No. 1
|8.00|22.67|17.20|5.50|20.00|9.20|6.50|8.25|14.20|24.50|
Individuo No. 2
|67.00|15.00|12.57|6.11|12.50|11.29|3.89|8.30|6.43|5.67|
Individuo No. 3
|41.50|30.67|13.50|44.00|39.00|0.00|13.57|11.67|7.56|3.80|
Individuo No. 4
|34.00|1.80|8.78|0.57|22.50|9.80|25.00|7.30|38.50|7.44|
Individuo No. 5
|4.89|5.00|10.22|18.33|3.30|23.67|98.00|99.00|78.00|1.14|
Individuo No. 6
|7.00|8.29|4.50|8.00|12.38|2.13|16.75|8.57|19.00|5.00|
Individuo No. 7
|9.29|7.60|1.67|44.00|18.00|4.40|12.57|9.33|33.00|58.00|
Individuo No. 8
|5.25|4.43|10.17|11.43|0.86|9.67|6.75|24.00|5.67|2.00|
Individuo No. 9
|2.25|1.33|3.50|20.00|45.50|6.00|1.67|23.00|2.67|20.50|
Individuo No. 10
|15.83|10.75|16.50|22.50|0.22|28.33|8.50|3.78|22.33|15.40|


"C:\Users\rubl3\Dropbox\8 semestre\Genetic Algorithms\P
Individuo No. 1
|76|93|38|33|81|64|21|97|74|42|
Individuo No. 2
|18|46|13|22|50|84|91|64|28|60|
Individuo No. 3
|34|85|47|1|16|12|84|76|48|68|
Individuo No. 4
|12|75|68|3|58|99|76|73|97|55|
Individuo No. 5
|14|16|1|79|95|82|22|98|41|47|
Individuo No. 6
|37|84|58|60|38|43|43|64|81|36|
Individuo No. 7
|88|11|5|57|7|17|53|4|59|28|
Individuo No. 8
|41|26|95|20|93|68|15|13|32|38|
Individuo No. 9
|94|18|60|35|37|57|37|61|55|58|
Individuo No. 10
|43|29|30|77|11|80|60|59|15|99|

Conclusiones

Debido a que la práctica tenía instrucciones un poco ambiguas, considero que la realización de esta practica pudo ser mejor. Sin embargo, si estuviera mal solo seria cambiar unas pequeñas cosas ya que la realización está bien hecha solo seria ajustar los valores o parámetros.

Código

```
#include <stdio.h>
#include<stdlib.h>
#include<string.h>
#include <iostream>
#include <time.h>

using namespace std;

int xorOp(int a, int b);
int main()
{
    int resp;
    do{

                  printf("1.-Binario \n 2.-Gray\n 3.-Real\n 4.-Entero\n
5-SALIR\n");
             scanf("%d",&resp);
             srand (time(NULL));
                   switch(resp)
                   {
                case 1:
                        system("cls");
                        int binario[10];
                            for (int j = 0; j < 10; j++)
                            {
                                cout << "Individuo No. " << j+1 << endl;
                                for (int i = 0; i < 10; i++)
                                        binario[i] = rand() % 2;

                                for (int i = 0; i < 10; i++)
                                {
                                    if(i == 0)
                                        printf("\t|%d|", binario[i]);
                                    else if(i < 9){
                                        printf("%d|", binario[i]);
                                    }
                                    else
                                        printf("%d|\n", binario[i]);
                                }
                                printf("\n");
                            }
                break;
                case 2:
                        int binario1[10];
                        int gray[10];
```

```
                        for (int j = 0; j < 10; j++)
                            {
                                cout << "Individuo No. " << j+1 << endl;
                                for (int i = 0; i < 10; i++)
                                        binario1[i] = rand() % 2;

                                gray [0] = binario1[0];

                                for (int i = 1; i < 10; i++)
                                {
                                    gray[i] = xorOp(binario1[i-1],
binario1[i]);
                                }
                                for (int i = 0; i < 10; i++)
                                {
                                    if(i == 0)
                                        printf("\t|%d|", binario1[i]);
                                    else if(i < 9){
                                        printf("%d|", binario1[i]);
                                    }
                                    else
                                        printf("%d|\n", binario1[i]);
                                }

                                for (int i = 0; i < 10; i++)
                                {
                                    if(i == 0)
                                        printf("\t|%d|", gray[i]);
                                    else if(i < 9){
                                        printf("%d|", gray[i]);
                                    }
                                    else
                                        printf("%d|\n", gray[i]);
                                }
                            }
                    break;
                    case 3:
                            float flotante[10];
                            for (int j = 0; j < 10; j++)
                            {
                                cout << "Individuo No. " << j+1 << endl;
                                for (int i = 0; i < 10; i++)
                                        flotante[i] = (float) (rand() %
(100))/((rand() % (10))+1);

                                for (int i = 0; i < 10; i++)
                                {
                                    if(i == 0)
                                        printf("\t|%.02f|", flotante[i]);
                                    else if(i < 9){
                                        printf("%.02f|", flotante[i]);
                                    }
                                    else
                                        printf("%.02f|\n", flotante[i]);
                                }
                                printf("\n");
```

```
                        }
                break;
                case 4:
                        int entero[10];
                        for (int j = 0; j < 10; j++)
                        {
                            cout << "Individuo No. " << j+1 << endl;
                            for (int i = 0; i < 10; i++)
                                entero[i] =rand() % 100;

                            for (int i = 0; i < 10; i++)
                            {
                                if(i == 0)
                                    printf("\t|%d|", entero[i]);
                                else if(i < 9){
                                    printf("%d|", entero[i]);
                                }
                                else
                                    printf("%d|\n", entero[i]);
                            }
                            printf("\n");
                        }
                break;
                   }
             }while(resp!=5);
             return 0;
}

int xorOp(int a, int b){
      if(a == b)
            return 0;
      else
            return 1;
}
```