Instituto Politécnico Nacional

Escuela Superior de Computo

Algoritmos Genéticos

Práctica 7: Cruza para permutaciones

Hernández López Ángel Zait

2014080682

3CM5

Periodo: 2019/01

Planteamiento del problema
Realizar un programa que realice de una población dada, los diferentes tipos de cruza que son Orcer crossover, Partially Mapped Crossover, Position Based Crossover, Order Based Crossocer y Cycle Crossover

Introducción
Hay distintos tipos de cruza genética, en este caso veremos en cruza para permutaciones, a continuación, se explicará brevemente cada uno de los distintos tipos de cruza:

Order crossover:
Técnica propuesta por Davis, el cual consiste en una serie de pasos:

1. Seleccionar aleatoria mente una sub-cadena del padre uno.
2. Producir un hijo copiando la sub-cadena en las posiciones correspondientes a padre uno. Las posiciones restantes se dejan en blanco.
3. Borrar los valores que ya se encuentren en la sub-cadena de padre dos. La secuencia resultante contiene los valores faltantes.
4. Colocar los valores en posiciones no conocidas del hijo de izquierda a derecha.
5. Para obtener el segundo hijo se repiten los pasos del uno al cuatro, pero tomando ahora la sub-cadena de padre dos.

Partially Mapped Crossover:
Técnica propuesta por Goldberg y Lingle que tiene los siguientes pasos:

1. Elegir aleatoriamente dos puntos.
2. Intercambiar estos dos segmentos en los hijos que se generan.
3. El resto de las cadenas que conforman los hijos se obtienen haciendo mapeos entre los dos padres:
   a) Si un valor no está contenido en el segmento intercambiado, permanece igual.
   b) Si está contenido en el segmento intercambiado, entonces se sustituye por el valor que tenga dicho segmento en el otro padre.

Position-Based Crossover:
Técnica fue propuesta por Syswerda, tiene similitud con order crossover, los pasos son los siguientes:

1. Seleccionar al azar un conjunto de posiciones de padre uno, pueden ser no consecutivas.
2. Producir un hijo borrando de padre uno todos los valores, excepto aquéllos para que hayan sido seleccionados en el paso anterior.
3. Borrar los valores seleccionados del padre dos. La secuencia resultante de valores se usará para completar el hijo.
4. Colocar en el hijo los valores faltantes de izquierda a derecha, de acuerdo a la secuencia de padre dos.
5. Repetir los pasos del uno al cuatro, pero tomando ahora la secuencia de padre dos.

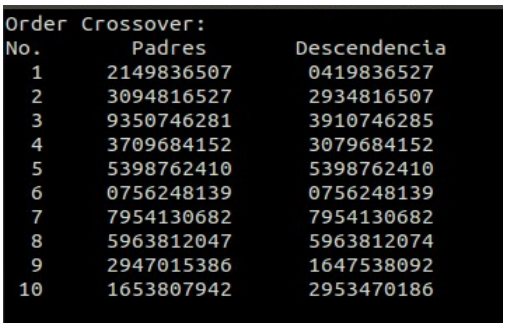
Order-based Crossover:
Técnica propuesta por Syswerda, y es una variante de la cruza de Position-Based Crossover; en este caso, primero se selecciona una serie de valores de padre uno. Luego, movemos de padre dos esos valores, y a continuación generamos un hijo a partir de padre dos prima. Finalmente completamos el hijo con los valores de la secuencia obtenida de padre uno, insertando de izquierda a derecha.

Cycle Crossover:
Propuesta por Oliver, Smith y Holland; similar a Position-Based Crossover, pues toma algunos valores de un padre y selecciona los restantes del otro, los pasos son los siguientes:

1. Encontrar un ciclo que se define mediante las posiciones correspondientes de los valores entre los padres.
2. Copiar los valores de padre uno que sean parte del ciclo.
3. Borrar de padre dos los valores que estén en el ciclo.
4. Rellenar el hijo con los valores restantes de padre dos, sustituyendo de izquierda a derecha.
5. Repetir los pasos del uno al cuarto, usando ahora padre dos.

Contenido
Se encuentran una serie de funciones el cual realizan cada uno de los tipos de ciclos, a continuación, se presentarán algunos de ellos y se explicará brevemente:

void OrderCrossover(void): Se realiza el método de cruza, se selecciona una pequeña cadena y se va copiando, después de eso, se va a ir dato por dato, para ver que se va a copiar para generar al nuevo hijo.



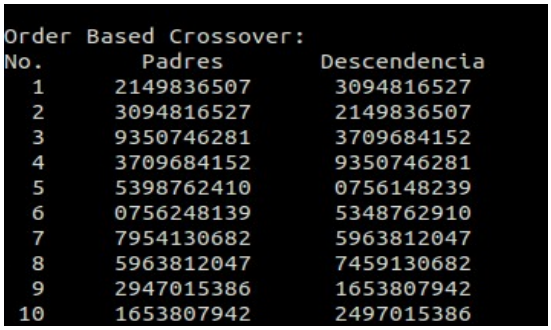
Order Crossover:

No.	Padres	Descendencia
1	2149836507	0419836527
2	3094816527	2934816507
3	9350746281	3910746285
4	3709684152	3079684152
5	5398762410	5398762410
6	0756248139	0756248139
7	7954130682	7954130682
8	5963812047	5963812074
9	2947015386	1647538092
10	1653807942	2953470186

void PositionBasedCrossover(void): Se seleccionan al azar los datos que se van a intercambiar, y se comparán con los datos del padre dos, para poder insertarlos en el hijo.

No.	Padres	Descendencia
1	2149836507	3140986527
2	3094816527	2149836507
3	9350746281	3750964182
4	3709684152	9735604281
5	5398762410	0758264139
6	0756248139	5396728410
7	7954130682	9653810472
8	5963812047	7954132068
9	2947015386	1965380742
10	1653807942	9457013862

void OrderBasedCrossover(void): Se tomo la base de la cruza anterior, pero en lugar de tener alelos del padre uno, ahora se tuvieron los del padre dos para hacer el hijo uno.



```
Order Based Crossover:
No.       Padres       Descendencia
  1     2149836507      3094816527
  2     3094816527      2149836507
  3     9350746281      3709684152
  4     3709684152      9350746281
  5     5398762410      0756148239
  6     0756248139      5348762910
  7     7954130682      5963812047
  8     5963812047      7459130682
  9     2947015386      1653807942
 10     1653807942      2497015386
```

Conclusiones

En esta practica se pudo observar otro tipo de cruza para distintos tipos de individuos, y que a pesar de ser otro tipo de dato, las cruzas son similares a los que se pueden hacer con una población con un tipo de cadenas de bits.