Introducción a la Computación Evolutiva

Dr. Carlos A. Coello Coello

Departamento de Computación

CINVESTAV-IPN

Av. IPN No. 2508

Col. San Pedro Zacatenco

México, D.F. 07300

email: ccoello@cs.cinvestav.mx

http: //delta.cs.cinvestav.mx/~ccoello

Técnicas de Cruza

- En los sistemas biológicos, la cruza es un proceso complejo que ocurre entre parejas de cromosomas. Estos cromosomas se alinean, luego se fraccionan en ciertas partes y posteriormente intercambian fragmentos entre sí.

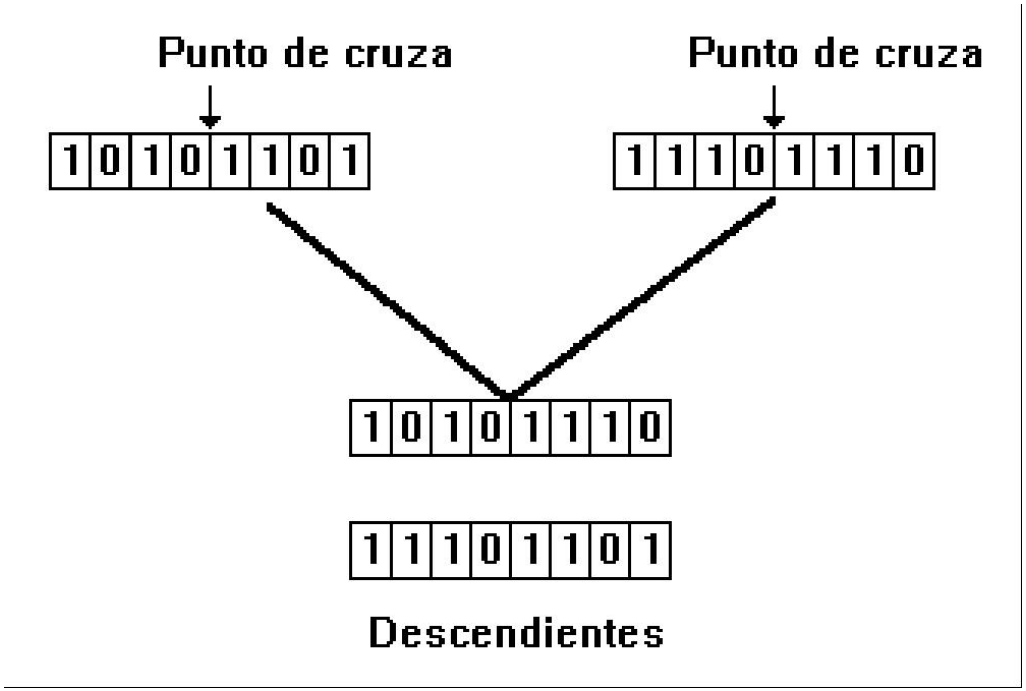
Técnicas de Cruza

- En computación evolutiva se simula la cruza intercambiando segmentos de cadenas lineales de longitud fija (los cromosomas).
- Aunque hemos visto técnicas de cruza básicas para representación binaria, éstas son generalizables a alfabetos de cardinalidad mayor, si bien en algunos casos requieren de ciertas modificaciones.

Técnicas de Cruza

- Comenzaremos por revisar las 3 técnicas básicas de cruza:
  1) Cruza de un punto
  2) Cruza de dos puntos
  3) Cruza uniforme

Cruza de un punto



Cruza de un punto

- Propuesta por Holland (1975).
- No suele usarse mucho en la práctica debido a sus inconvenientes. Puede demostrarse, por ejemplo, que hay varios esquemas que no pueden formarse bajo esta técnica de cruza.

Cruza de un punto

- Definamos a δ como la longitud de definición:
  $\delta(H)$ = distancia entre la primera y la última posición fija de un esquema H.

  Ejemplo:

  $\delta(*11*0*0*) = 7 - 2 = 5$
  $\delta(**1*****) = 0$

Cruza de un punto

- La cruza de un punto destruye esquemas en los que la longitud de definición es alta.
- Esto produce el denominado "sesgo posicional": los esquemas que pueden crearse o destruirse por la cruza dependen fuertemente de la localización de los bits en el cromosoma (Eshelman et al., 1989).

Cruza de un punto

- El problema fundamental de la cruza de un punto es que presupone que los bloques constructores son esquemas cortos y de bajo orden, y cuando esto no sucede (p.ej., con cadenas largas), suele no proporcionar resultados apropiados.

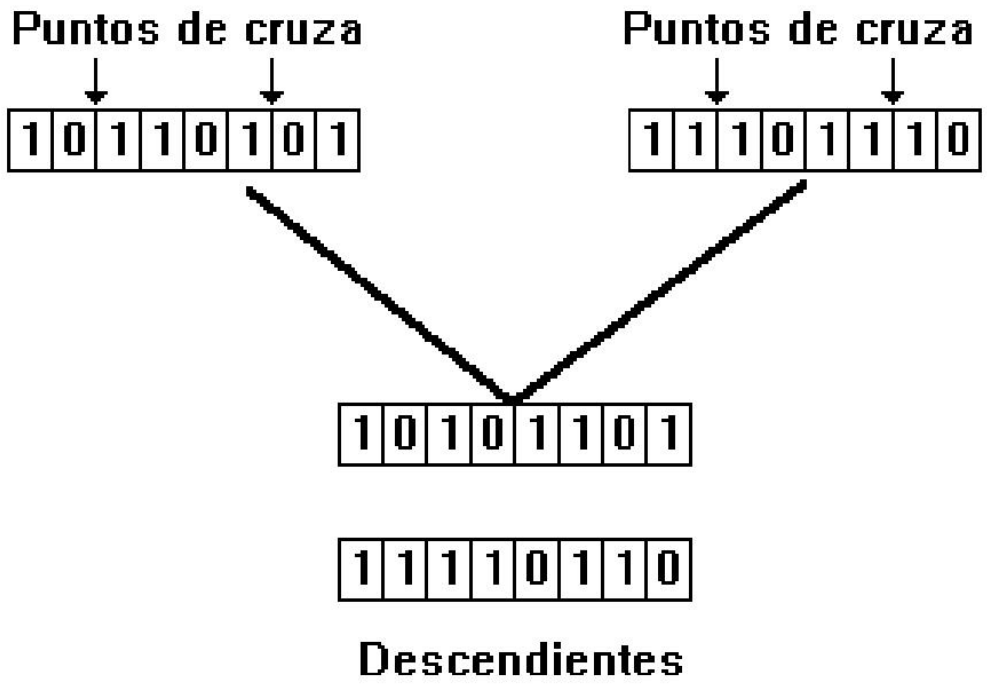
Cruza de un punto

- Obviamente, las aplicaciones del mundo real suelen requerir cadenas largas.
- La cruza de un punto trata también preferencialmente algunas posiciones del cromosoma, como por ejemplo los extremos de una cadena.

Cruza de un punto

- La cruza de un punto suele preservar también los "hitchhikers", que son bits que no son parte del esquema deseado, pero que debido a su similitud con ellos gozan de los beneficios de la cruza.

Cruza de dos puntos



Cruza de dos puntos

- De Jong (1975) fue el primero en implementar una cruza de n puntos, como una generalización de la cruza de un punto.
- El valor $n = 2$ es el que minimiza los efectos disruptivos (o destructivos) de la cruza y de ahí que sea usado con gran frecuencia.

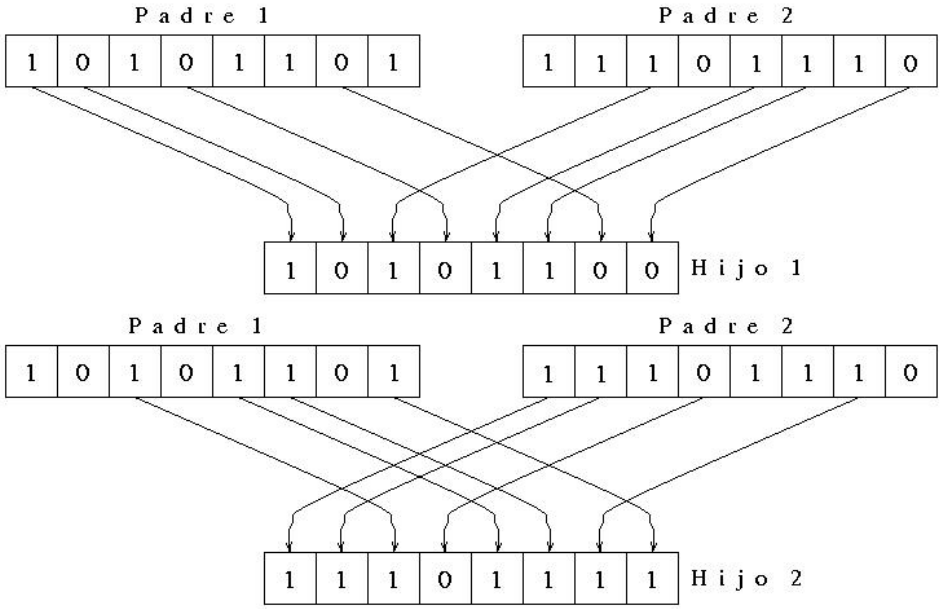
Cruza de dos puntos

- No existe consenso en torno al uso de valores para n que sean mayores o iguales a 3.
- Los estudios empíricos al respecto (De Jong, 1975; Eshelman et al., 1989) proporcionan resultados que no resultan concluyentes respecto a las ventajas o desventajas de usar dichos valores.

Cruza de dos puntos

- En general, sin embargo, es aceptado que la cruza de dos puntos es mejor que la cruza de un punto.
- Asimismo, el incrementar el valor de n se asocia con un mayor efecto disruptivo de la cruza.

Cruza uniforme



Cruza uniforme

- Propuesta originalmente por Ackley (1987), aunque se le suele atribuir a Syswerda (1989).
- En este caso, el número de puntos de cruza no se fija previamente.
- La cruza uniforme tiene un mayor efecto disruptivo que cualquiera de las 2 anteriores.

Cruza uniforme

- Suele usarse con $Pc = 0{,}5$.
- Algunos investigadores, sin embargo, sugieren usar valores más pequeños de Pc (Spears & De Jong, 1991).
- Cuando se usa $Pc = 0{,}5$, hay una alta probabilidad de que todo tipo de cadena binaria de longitud L sea generada como máscara de copiado de bits.

Cruza Acentuada

- Propuesta por Schaffer y Morishima (1987).
- En vez de calcular directamente la máscara (o patrón) de cruza, la idea es usar una cadena binaria de "marcas" para indicar la localización de los puntos de cruza.
- La idea fue sugerida por Holland (1975), aunque en un sentido distinto.

Cruza Acentuada

- La información extra se agrega al cromosoma de manera que el número y localizaciones de los puntos de cruza pueda ser objeto de manipulación por el AG.
- Por tanto, las cadenas tendrán una longitud del doble de su tamaño original.

Cruza Acentuada

- Marcamos con "1" las posiciones donde hay cruza y con "0" las posiciones donde no la hay.
- Se suelen usar signos de admiración para facilitar la escritura de las cadenas.

Cruza Acentuada

Ejemplo:

Cromosoma:

$$0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0 : 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0$$

cadena original L = 10 puntos de cruza L = 10

Puede interpretarse como:

01!100!01100

↑ ↑

Aquí se efectúa la cruza

Cruza Acentuada

Algoritmo:

- Copiar los bits de cada padre hacia sus hijos, de uno en uno.
- En el momento en que se encuentra un signo de admiración en cualquiera de los padres, se efectúa la cruza (es decir, se invierte la procedencia de los bits en los hijos).
- Cuando esto ocurre, los signos de admiración se copian también a los hijos, justo antes de que la cruza se efectúe.

Cruza Acentuada

Ejemplo:

Antes de la cruza:

P1 = a a a a a a a! b b b b b b b

P2 = c c c c! d d d d d d! e e e e

Después de la cruza:

H1 = a a a a d d d b b b e e e e

H2 = c c c c! a a a! d d d! b b b b

Cruza Acentuada

- Sólo se usa la primera parte de la cadena para calcular la aptitud, pero se espera que la selección, cruza y mutación tengan un efecto positivo sobre los puntos de cruza.
- La mutación actúa sobre los dos segmentos cromosómicos.
- Las probabilidades de que aparezcan unos en el segundo segmento se determinan de manera distinta a las del primero.

Cruza Acentuada

- La técnica reportó buenos resultados en un pequeño conjunto de funciones de prueba.
- Sin embargo, no hay evidencia contundente acerca de su efectividad.
- Tiene una buena inspiración biológica, porque estas marcas de cruza efectivamente existen en la naturaleza y se co-evolucionan junto con los cromosomas.

Sesgos de la Cruza

- El "sesgo" de la cruza se refiere a las tendencias de este operador hacia favorecer o no un cierto tipo de búsqueda.
- La búsqueda aleatoria es la única que no presenta ningún tipo de sesgo.
- Desde hace algún tiempo, se ha determinado que se requiere de algún tipo de sesgo para que una técnica de búsqueda sea efectiva (Mitchell, 1980).

Sesgos de la Cruza

- En algoritmos genéticos, se suelen considerar 2 tipos de sesgo para la cruza:
  1) Distribucional
  2) Posicional

Sesgos de la Cruza

- El **sesgo distribucional** se refiere al número de símbolos transmitidos durante una recombinación. Asimismo, se refiere a la medida en la que algunas cantidades tienen más tendencia a ocurrir que otras.

Sesgos de la Cruza

- El **sesgo distribucional** es importante porque está correlacionado con el número potencial de esquemas de cada padre que pueden ser recombinados por el operador de cruza.

Sesgos de la Cruza

- La cruza de un punto y la de dos puntos no tienen sesgo distribucional.
- La cruza de n puntos ($n > 2$) tiene un sesgo distribucional moderado.
- La cruza uniforme tiene un sesgo distribucional muy fuerte.

Sesgos de la Cruza

- El **sesgo posicional** caracteriza en qué medida la probabilidad de que un conjunto de símbolos se transmitan intactos durante la recombinación depende de las posiciones relativas de los mismos en el cromosoma.

Sesgos de la Cruza

- El **sesgo posicional** es importante porque indica qué esquemas es más probable que se hereden de padres a hijos.
- También indica la medida en la que estos esquemas aparecerán en nuevos contextos.

Sesgos de la Cruza

- La cruza de un punto tiene un fuerte sesgo posicional.
- Todo parece indicar, que la cruza de n puntos tiene también un sesgo posicional fuerte, aunque éste varía en función de n.
- La cruza uniforme no tiene sesgo posicional.

Variantes de la Cruza

- En la práctica, diversos aspectos de la cruza suelen modificarse para mejorar su desempeño.
- Una variante, por ejemplo, consiste en retener sólo a uno de los dos hijos producidos por una cruza sexual.
- Holland (1975) describe una técnica de este tipo.

Variantes de la Cruza

- Estudios empíricos han mostrado, sin embargo, que retener a los 2 hijos producidos por una cruza sexual reduce sustancialmente la pérdida de diversidad en la población (Booker, 1982).

Variantes de la Cruza

- Otra variante muy común es la de restringir los puntos de cruza a aquellas posiciones en las que los padres difieran.
- A esta técnica se le conoce como **sustitución reducida** (Booker, 1987).
- El objetivo es mejorar la capacidad de la cruza para producir hijos que sean distintos a sus padres.

Variantes de la Cruza

- Otra variante interesante es la llamada **cruza con barajeo** (Eshelman et al., 1989).
- En este caso, se aplica un operador de permutación a una parte de las cadenas de los padres antes de efectuar la cruza.
- Después de la cruza, se aplica la permutación inversa a fin de restaurar el orden original de los bits.

Variantes de la Cruza

- La **cruza con barajeo** tiene como objeto contrarrestar la tendencia de la cruza de n puntos ($n >= 1$) a causar con más frecuencia disrupción en los conjuntos de bits que están dispersos que en los que están juntos.

Cruza Segmentada

- Propuesta por Eshelman et al. (1989).
- Es una variante de la cruza de n puntos en la cual el número de puntos de cruza no es constante.
- Se usa una probabilidad s de que una subcadena tenga su extremidad derecha en una cierta posición subsecuente a su inicio.

Cruza Segmentada

- Iniciando de la primera posición del cromosoma (la posición la denotaremos con i), se genera aleatoriamente un número real $q \in [0, 1]$ y un número natural j tal que:

$$i < j <= L$$

L es la longitud de la cadena

Cruza Segmentada

- El valor q es considerado como la probabilidad de aceptar a j como un punto de cruza.
- Dependiendo de la relación entre s y q, el punto j puede o no ser aceptado. De esta forma, el número de puntos de cruza varía.
- Usualmente, se acepta j como un punto de cruza si se cumple que $q <= s$.

Cruzas con varios padres

- Aunque no son comunes en los algoritmos genéticos, existen también operadores de cruza que usan varios padres. Por ejemplo:
  - **Multi-parent uniform crossover** (Furuya & Haftka, 1993)
  - **Diagonal crossover** (Eiben et al., 1995)
  - **Scanning crossover** (Eiben et al., 1995)

Formas de Apareamiento

- Otro punto interesante a considerar son las técnicas de apareamiento (es decir, quién puede recombinarse con quién).
- A continuación revisaremos rápidamente las formas más comunes de apareamiento, de acuerdo a Goldberg (1989).

Formas de Apareamiento

- **Random Mating** (aleatorio)
  Se eligen los individuos aleatoriamente, con la misma probabilidad.
- **Inbreeding** (entre parientes)
  Se recombinan individuos similares
- **Line Breeding** (semental)
  Un solo super-individuo (aptitud alta) se recombina con una población base y sus hijos se seleccionan como padres.

Formas de Apareamiento

- **Outbreeding** (entre desconocidos)
  Sólo se recombinan individuos muy diferentes.
- **Self-fertilization** (auto-fertilización)
  Un individuo se recombina con sí mismo.
- **Cloning** (clonación)
  Un individuo se copia sin modificaciones

Formas de Apareamiento

- **Positive assorting mating**
  Se recombinan individuos similares.
- **Negative assorting mating**
  Se recombinan individuos diferentes.

Comportamiento Deseable de la Cruza

- Todos estos operadores descritos anteriormente, siguen el principio Mendeliano de la herencia: cada gene que tiene un hijo, es una copia de un gene heredado de alguno de sus padres.
- Cabe mencionar, sin embargo, que esto no tiene que ser así en computación evolutiva.

Comportamiento Deseable de la Cruza

- Algunos investigadores han destacado que el énfasis de la cruza debe ser el poder generar todas las posibles combinaciones de bits (de longitud L) que hayan en el espacio de búsqueda del problema (Radcliffe, 1991).

Comportamiento Deseable de la Cruza

- Dada una cierta representación binaria, ni la cruza de un punto, ni la de n puntos son capaces de lograr esto (generar cualquier combinación de bits posible).
- La cruza uniforme, sin embargo, sí puede hacerlo.
- Algunos investigadores han propuesto otras variantes de la cruza motivados por este problema.

Comportamiento Deseable de la Cruza

- Radcliffe (1991) propuso una técnica denominada **recombinación respetuosa aleatoria**.
- Según esta técnica, se genera un hijo copiando los bits en los que sus padres son idénticos, y eligiendo luego, valores al azar para llenar las posiciones siguientes.
- Si se usan cadenas binarias, y $Pc = 0.5$, la cruza uniforme es equivalente a esta recombinación.

Cruza para representaciones alternativas

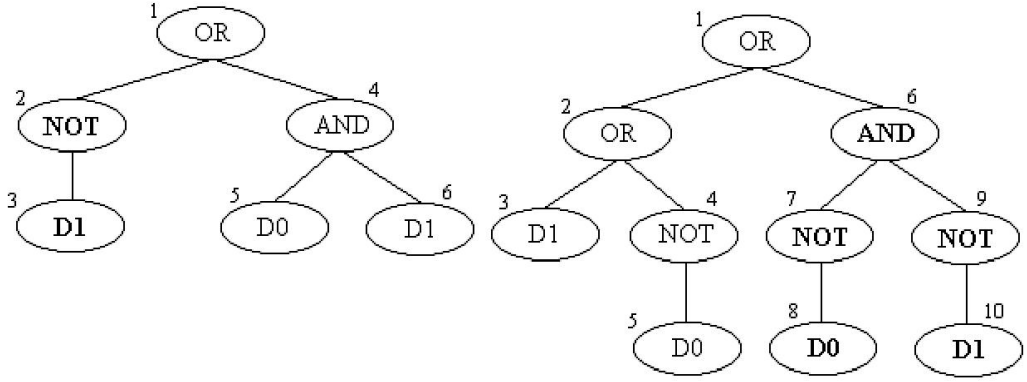
- Programación Genética
- Permutaciones
- Representación real

Programación Genética

- Al usarse representación de árbol, la cruza sigue funcionando de manera muy similar a la cruza convencional, sólo que en este caso, se intercambian sub-árboles entre los 2 padres.
- El primer hijo se produce borrándole al primer padre el fragmento indicado por el punto de cruza e insertando el fragmento (sub-árbol) correspondiente del segundo padre.
- El segundo hijo se produce de manera análoga.

Programación Genética

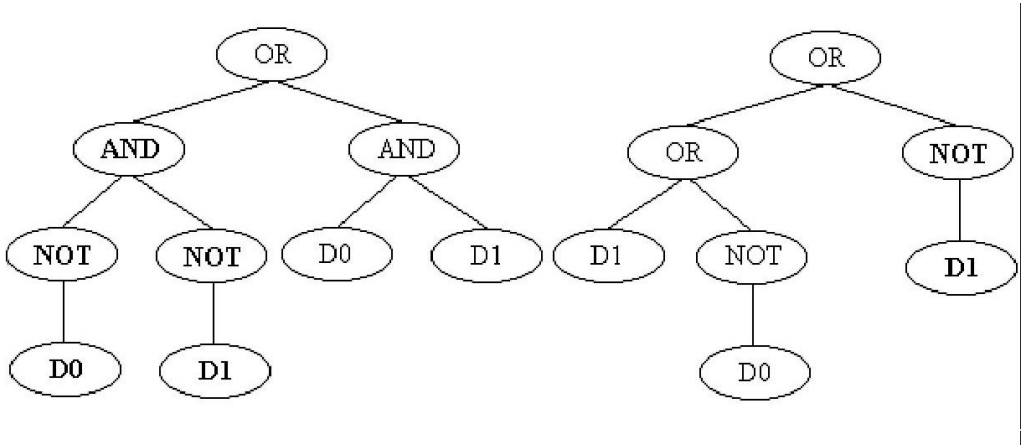
- Ejemplo: Dados los 2 padres siguientes



Las expresiones S de estos 2 padres son:
(OR (NOT D1) (AND D0 D1)) y
(OR (OR D1 (NOT D0)) (AND (NOT D0) (NOT D1)))

Programación Genética

- Si el punto de cruza del primer padre es 2, y el del segundo es 6, entonces los hijos resultantes serán:



Programación Genética

- Observaciones:

  - Tipicamente, los 2 padres serán de tamaños distintos.
  - Los padres son seleccionados mediante alguna de las técnicas que vimos antes.
  - Suele limitarse la profundidad máxima de un árbol.
  - La raíz es un punto de cruza válido.

Cruza para Permutaciones

- Al efectuar cruza entre 2 cadenas que usan representación de permutaciones, los hijos invariablemente serán no válidos.
- La representación de permutaciones se usa frecuentemente en problemas de optimización combinatoria, como el del viajero.
- Ejemplo de una permutación: 1 2 3 4 5 6 7 8 9

Cruza para Permutaciones

- Order Crossover
- Partially Mapped Crossover
- Position-Based Crossover
- Order-Based Crossover
- Cycle Crossover
- Otros

Cruza para Permutaciones

- **Order Crossover** (OX):Propuesta por Davis (1985).
  El algoritmo es el siguiente (los padres son P1 y P2):

  1) Seleccionar (aleatoriamente) una sub- cadena P1.

  2) Producir un hijo copiando la sub-cadena en las posiciones correspondientes a P1.
     Las posiciones restantes se dejan en blanco.

Cruza para Permutaciones

- Algoritmo de **Order Crossover** (continúa):

3) Borrar los valores que ya se encuentren en la sub-cadena de P2. 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 1 al 4, pero tomando ahora la sub-cadena de P2.

Cruza para Permutaciones

Ejemplo de **Order Crossover**:

$$P1 = 9 \quad 8 \quad 4 \quad 5 \quad 6 \quad 7 \quad 1 \quad 2 \quad 3 \quad 10$$

$$P2 = 8 \quad 7 \quad 1 \quad 2 \quad 3 \quad 10 \quad 9 \quad 5 \quad 4 \quad 6$$

Sub-cadena elegida: 5 6 7 1 (de P1) Primer hijo:

$$H1 = X \quad X \quad X \quad 5 \quad 6 \quad 7 \quad 1 \quad X \quad X \quad X$$

Cruza para Permutaciones

Ejemplo (continúa):
Borrar de P2 la sub-cadena tomada de P1:

$$P2' = 8 \quad X \quad X \quad 2 \quad 3 \quad 10 \quad 9 \quad X \quad 4 \quad X$$

Determinar los valores faltantes de H1 sustituyendo (de izquierda a derecha) los valores que aparecen en P2':

$$H1 = 8 \quad 2 \quad 3 \quad 5 \quad 6 \quad 7 \quad 1 \quad 10 \quad 9 \quad 4$$

Para obtener H2, el procedimiento es similar, aunque ahora la sub-cadena se tomará de P2 y la sustitución se hará a partir de P1'.

Cruza para Permutaciones

- **Partially Mapped Crossover** (PMX): Propuesta por Goldberg y Lingle (1985).Tiene ciertas similitudes con OX. El algoritmo es el siguiente:

  1) Elegir aleatoriamente dos puntos de cruza.

  2) Intercambiar estos 2 segmentos en los hijos que se generan (como la cruza de 2 puntos convencional).

Cruza para Permutaciones

- **Algoritmo de Partially Mapped Crossover** (continúa):

  3) El resto de las cadenas que conforman los hijos se obtienen haciendo mapeos entre los 2 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.

Cruza para Permutaciones

- Ejemplo de **Partially Mapped Crossover**:

$$P1 = 9\ 8\ 4\ \mid\ 5\ 6\ 7\ \mid\ 1\ 3\ 2\ 10$$

$$P2 = 8\ 7\ 1\ \mid\ 2\ 3\ 10\ \mid\ 9\ 5\ 4\ 6$$

Los hijos son:

$$H1 = X\ X\ X\ \mid\ 2\ 3\ 10\ \mid\ X\ X\ X\ X$$

$$H2 = X\ X\ X\ \mid\ 5\ 6\ 7\ \mid\ X\ X\ X\ X$$

Cruza para Permutaciones

Ejemplo de **Partially Mapped Crossover** (continúa):
Para completar $H1$ y $H2$, copiamos primero los valores que no están en el segmento intercambiado:

$$H1 = 9\ 8\ 4\ \mid\ 2\ 3\ 10\ \mid\ 1\ X\ X\ X$$

$$H2 = 8\ X\ 1\ \mid\ 5\ 6\ 7\ \mid\ 9\ X\ 4\ X$$

Ahora mapeamos los valores restantes:

$$H1 = 9\ 8\ 4\ 2\ 3\ 10\ 1\ 6\ 5\ 7$$

$$H2 = 8\ 10\ 1\ 5\ 6\ 7\ 9\ 2\ 4\ 3$$

Cruza para Permutaciones

- **Position-based Crossover**: Propuesta por Syswerda (1991) como una adaptación de la cruza uniforme para permutaciones. El algoritmo es el siguiente:

  1) Seleccionar (al azar) un conjunto de posiciones de $P1$ (no necesariamente consecutivas).

  2) Producir un hijo borrando de $P1$ todos los valores, excepto aquéllos que hayan sido seleccionados en el paso anterior.

Cruza para Permutaciones

3) Borrar los valores seleccionados de P2. 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 P2.

5) Repetir los pasos del 1 al 4, pero tomando ahora la secuencia de P2.

Cruza para Permutaciones

Ejemplo de Position-based Crossover:

$$P1 = 9\ 8\ 4\ 5\ 6\ 7\ 1\ 2\ 3\ 10$$

$$P2 = 8\ 7\ 1\ 2\ 3\ 10\ 9\ 5\ 4\ 6$$

Valores elegidos de $P1$: 8 6 2 10

Producir un hijo:

$$H1 = X\ 8\ X\ X\ 6\ X\ X\ 2\ X\ 10$$

Cruza para Permutaciones

Borrar de P2 la secuencia usada para H1:

$$P2' = X\ 7\ 1\ X\ 3\ X\ 9\ 5\ 4\ X$$

Sustituir de izquierda a derecha los valores que aparecen en P2':

$$H1 = 7\ 8\ 1\ 3\ 6\ 9\ 5\ 2\ 4\ 10$$

Para obtener H2, el procedimiento es similar, pero la secuencia se toma ahora de P2 y la sustitución se hace a partir de P1'.

Cruza para Permutaciones

- **Order-based Crossover**: Propuesta por Syswerda (1991) como una ligera variante de **Position-based Crossover** en la que se cambia el orden de los pasos del algoritmo.
- En este caso, primero seleccionamos una serie de valores de $P1$. Luego, removemos de $P2$ esos valores. A continuación generamos un hijo a partir de $P2'$.
- Finalmente, completamos el hijo con los valores de la secuencia obtenida de $P1$ (insertada de izquierda a derecha en el orden impuesto por $P1$).

Cruza para Permutaciones

Ejemplo de **Order-based Crossover**:

$$P1 = 9\ 8\ 4\ 5\ 6\ 7\ 1\ 2\ 3\ 10$$

$$P2 = 8\ 7\ 1\ 2\ 3\ 10\ 9\ 5\ 4\ 6$$

Valores elegidos de P1: 8 6 2 10

Removamos de P2 estos valores:

$$P2' = X\ 7\ 1\ X\ 3\ X\ 9\ 5\ 4\ X$$

Cruza para Permutaciones

Ejemplo de **Order-based Crossover** (continúa):
Producir un hijo:

$$H1 = X\ 7\ 1\ X\ 3\ X\ 9\ 5\ 4\ X$$

Insertamos ahora la secuencia elegida de P1:

$$H1 = 8\ 7\ 1\ 6\ 3\ 2\ 9\ 5\ 4\ 10$$