Introducción a la Computación Evolutiva

Dr. Carlos Artemio Coello Coello
Tarea No. 4
22 de junio de 2022

Optimización Combinatoria

1. El objetivo de esta tarea es que implemente un algoritmo genético con una representación de permutaciones, el cual se utilizará para resolver un problema clásico de optimización combinatoria: el problema de la asignación cuadrática.

Para fines de esta tarea, el tipo de técnica de cruza de permutaciones que deberá implementar será la indicada a continuación:

- **André Fabián** y **Rogelio** : *Partially Mapped Crossover* (PMX).
- **Estefanía** : *Position-based Crossover*.
- **Alberto Josué** : *Order-based Crossover*.

El algoritmo genético implementado deberá tener las siguientes características:

- **Remoción de duplicados**: En la generación cero y en las generaciones posteriores (tras haber generado a través de la selección, cruza, mutación e inversión a la siguiente población), deberá checarse que no existan duplicados (o sea, no deben existir en la misma población dos individuos que codifiquen la misma permutación). Los duplicados deberán reemplazarse por un individuo generado al azar o producto de una alteración del individuo a removerse (por ejemplo, una mutación).
- **Mutación**: El tipo de mutación que debe implementar es el que le corresponda de acuerdo a la lista siguiente:
  - **Estefanía** y **Alberto Josué**: *Mutación por Intercambio Recíproco*.
  - **Rogelio**: *Mutación por Desplazamiento*.
  - **André Fabián**: *Mutación por Inserción*.
- **Selección**: Deberá implementarse la selección mediante torneo binario determinístico, usando barajeo de la población.
- **Representación**: Debe implementarse una representación de permutaciones (o sea, cadenas de enteros de 0 a $n-1$, donde n representa el tamaño de la matriz de entrada, de tal forma que la secuencia numérica no tiene ningún elemento repetido).

- **Inversión**: Debe implementarse un operador de inversión que usará los siguientes parámetros: porcentaje de aplicabilidad (como en la cruza y la mutación), punto inicial y punto final. La política de reemplazo es opcional, pero debe aclararla en su reporte. El operador se aplicará siempre después de la cruza y la mutación, pero debe decidirse si se reemplazará siempre al individuo original, si será en cierto número (aleatorio) de veces, o si se hará cuando el individuo tras inversión tenga mejor aptitud que el original.

Lo que debe entregarse es lo siguiente:

1. (**80 puntos**) El código fuente en C/C++ de un programa que implemente un algoritmo genético con codificación de permutaciones, utilizando inversión, cruza y mutación (como se indicó anteriormente). En los comentarios de su programa y en el reporte correspondiente deberá indicar claramente los detalles de los operadores implementados. El código fuente deberá tener suficientes comentarios como para hacerlo fácilmente comprensible y deberá incluirse en un CD en donde también deberá ir el ejecutable correspondiente, así como TODAS las corridas que haya efectuado. Los parámetros en general que utilice quedan a su libre elección, pero deberán mencionarse claramente en los comentarios del programa y en el reporte correspondiente. De ser necesario, el autor deberá demostrar el programa personalmente al instructor. El programa deberá pedir al usuario (de forma interactiva y NO a través de la línea de comandos) los parámetros principales del algoritmo genético: tamaño de la población, porcentaje de cruza, porcentaje de mutación, porcentaje de inversión, número máximo de generaciones y nombre del archivo donde se almacenarán los datos de cada corrida. En cada generación deberá retenerse a la mejor solución (elitismo), y deberán imprimirse al menos las siguientes estadísticas por generación: media de aptitud de la población, aptitud máxima y mínima, número total de cruzas efectuadas, número total de mutaciones efectuadas, número total de inversiones efectuadas y permutación correspondiente al mejor individuo (el que tenga la aptitud más alta). Adicionalmente, deberá imprimirse el mejor individuo global (es decir, contando todas las generaciones), su aptitud, la permutación que codifica y su costo.

2. (**60 puntos**) Una corrida de ejemplo por cada uno de los problemas incluidos en el punto 4, y los resultados promediados de AL MENOS 20 corridas independientes para cada problema en las que se usen los mismos parámetros (porcentaje de cruza, mutación e inversión, tamaño de la población y máximo número de generaciones) pero diferente semilla para generar números aleatorios. En el reporte deberá aparecer una corrida representativa por cada problema. También se deberá incluir una tabla que incluya: la permutación correspondiente a la mejor solución final de cada una de las 20 corridas efectuadas, su costo y su aptitud (en caso de que difieran estos 2 valores). Luego, en la parte inferior de la tabla se debe reportar (con respecto a las 20 corridas): la mejor solución obtenida, la media de aptitud, la peor solución obtenida y la desviación estándar de las 20 corridas efectuadas. Deberán mostrarse las permutaciones (especificando claramente si deben leerse de izquierda a derecha o viceversa) y sus costos correspondientes

en cada caso. Asimismo, se incluirá la gráfica de convergencia correspondiente a la solución que esté en la mediana de los resultados obtenidos (de las 20 corridas) para cada problema. En esta gráfica se mostrará en el eje de las x el número de generaciones (de cero al máximo permisible) y en el eje de las y se mostrará la aptitud máxima de cada generación. TODAS las corridas efectuadas deberán aparecer en archivos diferentes en un CD, además del código fuente del programa.

3. (**Bonificación 1: hasta 40 puntos**) Se bonificará con diferentes cantidades (totalizando un máximo de 40 puntos) a los que presenten al menos una corrida en la que se iguale el óptimo proporcionado en este documento para cada una de las 3 funciones de prueba especificadas. Esta corrida (o corridas) se deberá presentar en el CD, indicando claramente los parámetros usados para el AG, así como la semilla de aleatorios empleada. Los resultados correspondientes (indicando claramente la permutación cuyo costo iguala el del óptimo) deberán incluirse en el reporte y en el CD (indique el nombre de los archivos correspondientes y grábelos en un directorio llamado **óptimos**). Las bonificaciones son las siguientes:

`tai12.dat`	Bonificación: **5 puntos**
`tai15.dat`	Bonificación: **10 puntos**
`tai30.dat`	Bonificación: **25 puntos**

4. (**Bonificación 2: 20 puntos**) Se bonificará con un máximo de 20 puntos al que incluya en su implementación operadores distintos de los vistos en clase y que logren mejorar el desempeño de su algoritmo genético (esto debe validarse efectuando al menos 20 corridas con los nuevos operadores y se deberá respaldar con una discusión en la que se haga ver que la mejora es significativa). Es válido recurrir a la literatura especializada para recurrir a otros operadores pero, en ese caso, deberá citarse la fuente correspondiente. Si el operador es de la autoría del que realizó la tarea, deberá discutirse a detalle el funcionamiento del mismo, incluyendo su motivación de diseño.

5. **Funciones de prueba**: Se le proporcionarán 4 archivos de prueba para evaluar el desempeño de su programa. Estos archivos estarán disponibles en la página web del curso, y se llaman: **ajuste.dat**, **tai12.dat**, **tai15.dat**, **tai30.dat**. El primero de ellos es para ajustar su programa, y los otros 3 son los ejemplos que debe utilizar para su reporte. El formato de los archivos es el siguiente:

   `tamaño de la matriz` (n)

   `matriz de distancias` (dist[n][n])

   `matriz de flujos` (flujo[n][n])

El objetivo es minimizar el costo total, definido por:

$$\text{costo} = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} dist[i][j] \times flujo[x[i]][x[j]] \tag{1}$$

donde: $x[i]$ ($i = 0, \ldots, n-1$) se refiere a la permutación codificada en el algoritmo genético. Se le sugiere escribir una función (independiente) que calcule la aptitud de una permutación a partir de las matrices de distancias y flujos. Si lee los datos del archivo **ajuste.dat**, y proporciona la permutación: **2 3 4 0 1**, el costo debe ser **50** (valor óptimo).

Los costos óptimos para los demás archivos son los siguientes:

`tai12.dat`	Costo óptimo = 224416
`tai15.dat`	Costo óptimo = 388214
`tai30.dat`	Costo óptimo = 1818146

Fecha de entrega: **Viernes 1 de julio a las 12:00hrs**. Toda tarea entregada tarde será penalizada con 10% (sobre la calificación obtenida) por cada periodo de 24 horas que se retrase su entrega.