Guión de prácticas 3. Computación Evolutiva (CE)

Nesta práctica aplicaremos unha estratexia de busca de solucións baseada en computación evolutiva (CE) ao problema do "viaxante de comercio". A práctica divídese en dous apartados:

Algoritmo CE básico (7 puntos; especificación obrigatoria)

Neste apartado desenvolverase unha versión básica do algoritmo CE. Todos deberedes utilizar o mesmo deseño do algoritmo. As súas características son as seguintes:

1. **Representación** idéntica á empregada nas prácticas anteriores
2. **Poboación inicial**

   Tomaremos unha poboación de N_{pob}=100 individuos, que se inicializarán do seguinte xeito:

   - 50% da poboación mediante unha inicialización totalmente aleatoria das solucións.
   - 50% mediante unha estratexia voraz na que **se seleccionará aleatoriamente a primeira cidade** da representación e, a partir desta, as restantes seguirán o criterio voraz de mínima distancia.

3. **Función de *fitness*** (mesma función de custe das prácticas anteriores)

4. **Selección da poboación**

   A selección debe permitir que algún individuos de menor calidade entren a formar parte dunha solución. Aplicarase o método de **selección por torneo de tamaño 2**, no que se selecciona o individuo de mellor *fitness* entre k=2 individuos escollidos aleatoriamente de entre a poboación.

5. **Operador de cruce**

Empregaremos o operador "*order crossover*" (Transparencia 5.5.29; Libro Eiben, cap. 3.5, p. 54), que non precisa mecanismo de reparación. Neste operador, fíxanse aleatoriamente dous puntos de corte. O segmento entre eses dous puntos de P1 cópianse no fillo F1, nas mesmas posicións. A partir de aí, ciclicamente se enchen as posicións valeiras cos xens de P2 que non estean repetidos. Repítese a operación de xeito inverso para producir o fillo F2, partindo de P2 e completando cos xens de P1.

Guión de prácticas 3. Computación Evolutiva (CE)

Exemplo: P1: {1 2 3 **4 5 6 7** 8 9}

P2: {9 3 7 **8 2 6 5** 1 4}

F1: {_ _ _ **4 5 6 7** _ _} →{3 8 2 **4 5 6 7** 1 9}

F2: {_ _ _ **8 2 6 5** _ _} →{3 4 7 **8 2 6 5** 9 1}

Utilizaremos unha probabilidade de cruce p_c=0.9.

6. Operador de mutación

Usaremos o operador de **intercambio recíproco** (transparencia 5.5.33), no que dúas cidades **distintas** elixidas aleatoriamente intercambian as súas posicións

{1 **2** 3 4 5 6 7 8 **9**} → {1 **9** 3 4 5 6 7 8 **2**}

Utilizaremos unha probabilidade de mutación p_m=0.01. Este valor asegura que **en promedio** hai, aproximadamente, un intercambio por cada individuo.

7. Mecanismo de remprazo

O remprazo farase por un mecanismo xeracional simple con **elitismo** (dos **dous mellores**), seguindo o esquema da transparencia 5.5.36. Os N-2 descendentes con mellor *fitness* substituirán aos N-2 individuos con peor *fitness* da poboación actual (t). Visto isto, o mais práctico é xerar unha poboación intermedia (t+1) de tamaño N-2=98 descendentes que, unha vez cruzados e mutados, substituirán aos N-2 peores individuos da poboación (t).

8. Criterio de parada

Finalizaremos a busca cando se teñan xerado 1.000 poboacións.

Algoritmo CE mellorado (3 puntos; especificación voluntaria)

Nesta sección o obxectivo e realizar melloras ao algoritmo obtido previamente. Poderá modificarse calquera dos parámetros ou operadores da metaheurística:

- O número de individuos na poboación
- O operador de inicialización, modificando o peso da estratexia voraz fronte á aleatoria
- O operador de selección
- Os operadores de cruce ou mutación ou as probabilidades asociadas
- O mecanismo de remprazo

Guión de prácticas 3. Computación Evolutiva (CE)

- O criterio de parada
- Introducindo unha estratexia de reinicialización cando transcorridas un número determinado de iteracións non se obteñan melloras significativas na mellor das solucións.

Traza de execución e caso de uso

Estes tres requisitos son de atención obrigatoria:

1. O programa deberá poderse compilar en liña de comandos. A entrega deberá incluír un arquivo **LEEME.txt** que especifique a directiva de compilación

2. O programa deberá poder executarse en liña de comandos, coa sintaxe

a.out distancias.txt [aleatorios.txt]

Cando se utilicen os dous parámetros o programa deberá ler os números aleatorios que precise a partir dun arquivo de nome **aleatorios.txt**. Proporcionarase o arquivo de nome aleatorios.txt xunto cun caso de uso completo coa execución da metaheurística para eses números, no que se describirán os resultados a obter.

3. A saída por pantalla deberá ser idéntica (en contido e formato) á especificada na traza, para facilitar a detección de erros de implementación.

Entrega da práctica

A data límite de entrega será a que se especifique na ferramenta de entregas da Aula Virtual.

Será obrigatorio entregar o código fonte correspondente a cada unha das dúas partes (habilitarase unha entrega diferente para cada parte). Cada entrega consistirá nun arquivo comprimido, e en publicar os resultados de **10 execucións diferentes do algoritmo** nunha folla de cálculo compartida (da que proporcionaremos o enlace mediante un anuncio na Aula Virtual). Ademais, xunto co código da especificación voluntaria se incluirá un documento de texto (máximo media páxina) no que se explique en qué consistiron as melloras introducidas e unha valoración persoal de porqué as consideras apropiadas para o problema.

Avaliación da práctica

Este traballo contará para a parte práctica da materia cunha ponderación do 40%. A avaliación do funcionamento da especificación obrigatoria realizarase mediante un test de autoavaliación sobre os resultados de execución da vosa implementación en diferentes casos (matrices de distancias, números aleatorios, ...) en diferentes etapas do proceso de busca.
A avaliación da especificación voluntaria puntuará

- a porcentaxe de mellora conseguida sobre a solución inicial e a acadada na especificación obrigatoria
- a xustificación da adecuación das melloras introducidas