Algoritmos Genéticos – Problema de Optimización Simple

Objetivos

En esta sesión se pretende que el alumno practique en la solución de problemas reales en los que la aplicación de algoritmos de computación evolutiva simples son aplicables de forma directa.

En esta última entrega, se hará hincapié en que sea capaz de comparar variantes de algoritmos evolutivos para reconocer la más adecuada en un determinado problema.

Esto incluye tanto el estudio de las características típicas de este tipo de problemas como la solución de los mismos por medio de programas creados por el alumno.

Descripción del Problema

El problema propuesto consiste en generar un programa en python que permita resolver el problema de la distribución de vídeos en servidores de caché para mejorar el servicio a los usuarios. El problema se describe en el documento adjunto al enunciado.

Para realizar este programa, el alumno deberá emplear la librería *Distributed Evolutionary Algorithms in Python* (DEAP) para ayudarse en las partes más genéricas del algoritmo genético.

Se dispone de una ayuda completa de la librería en el siguiente enlace. Se puede consultar el tutorial más básico a cerca del funcionamiento de la librería en el siguiente enlace.

Trabajo del Alumno

Esta entrega pretende ser el resultado final de los ejercicios realizados hasta el momento. Por ello reutilizará en la entrega todo el código y documentación generada en las sesiones anteriores.

En sesiones anteriores se ha trabajado en:

1. Lectura de los datos del problema. Se nos presentan diferentes situaciones o variantes a un mismo problema (organización de una flotilla de coches/taxis) que debemos ser capaces de resolver de forma genérica, para que la solución no dependa de un determinado conjunto de vehículos, solicitudes de viaje o tamaño de la ciudad.
2. Se ha planteado una descripción interna en el programa que permita representar una posible configuración de distribuciones de pasajeros sobre los vehículos. Se ha implementado también una función que permite otorgar una puntuación sobre cada distribución según se adapte a las condiciones del problema.

Implementación

Se pide ahora al alumno completar una entrega final de resumen de todas las actividades anteriores, que adicionalmente nos permita comparar alguna variante sobre la solución básica planteada.

Para ello, se tomarán los resultados obtenidos en anteriores sesiones de prácticas y se completarán de la siguiente forma:

1. Se necesita incluir la funcionalidad por la que se almacenará la respuesta encontrada como la óptima al problema en un fichero de texto, tal y como se solicita en el desafío HashCode'18.

   Esto implica que se necesitará una funcionalidad que permita "traducir" de forma automática del genotipo (representación interna) al fenotipo (rep. externa) de los individuos.

   Parte de la evaluación del ejercicio dependerá de los resultados que se obtengan por este medio, así que se recomienda asegurarse de su correcto funcionamiento.

2. Se necesita incluir una funcionalidad que permita analizar el trabajo interno que ha realizado el algoritmo evolutivo. Para ello, se recogerán estadísticas de su funcionamiento y se generarán gráficas que permitan analizar ese comportamiento de forma sencilla. Existe un tutorial y los ejemplos de código ya facilitados (tener en cuenta que los ejemplos, las graficas son más sencillas de lo que se pide).

3. Se pide al alumno proponer 3 alternativas o configuraciones diferentes del esquema básico para estudiar el comportamiento de diferentes factores en el resultado final.

   Se pueden modificar:

   a) Esquemas de selección, cruce o mutación.

   b) Parámetros de cada uno de ellos: probabilidad de cruce, probabilidad de mutación, tamaño en los torneos, ...etc.

   c) Parámetros generales del algoritmo: tamaño de población, numero de generaciones,...

   d) Etc.

   Se deberá completar el informe final de la entrega incluyendo una comparativa que muestre los resultados obtenidos en cada variante.

4. A la hora de realizar la competición entre los grupos de la clase, se solicita que la solución incluya una forma sencilla de ejecutar la configuración que se considere como la más apropiada.

   Para simplificar la comparación, el código entregado ya tendrá preparada l**a mejor configuración** encontrada para ejecutar y el generar un fichero con la respuesta óptima (con el formato que se indica en el enunciado del HashCode'18) para el problema configurado en el **fichero `d_metropolis.in`**. El obtener el fichero de solución solo deberá implicar ejecutar el fichero **`main.py`**, que estará en la raíz del directorio que incluya el código fuente. Este fichero incluirá también un método sencillo de determinar sobre qué fichero de entrada se desea obtener una solución.

Informe

Incluye un fichero en formato PDF que incluya el contenido completo de los resultados obtenidos en las sesiones de prácticas anteriores.

Además se deberá completar con una comparativa de las variantes que se han estudiado en los algoritmos propuestos.

En cada variante, se deberán realizar diversos experimentos que permitan comprobar tendencias y se deberá tener en cuenta que existen varias situaciones del problema a resolver.

Si se quiere explorar la influencia de un parámetro en concreto, se deberán hacer varios experimentos, modificando ese parámetro para poder llegar a una conclusión. Ej: La conclusión: "A medida que se incrementa el valor del parámetro XX, la adaptación de la solución disminuye, en los conjuntos de datos de más de YY viajes" implica que se han comprobado un rango de valores para ese parámetro (no solo 2 valores) y se ha repetido ese experimento en varios conjuntos de datos.

Para obtener conclusiones más claras, es importante solo modificar un parámetro concreto en los experimentos, de manera que la modificación en el resultado no dependa de más de una variable.

En el informe, esta comparativa deberá incluir:

- Descripción de los ensayos que se han realizado
- Resultados numéricos y/o gráficos obtenidos en cada uno
- Conclusiones de los alumnos sobre dichos resultados

Recursos Necesarios

Se adjunta al enunciado:

- El repositorio de GitHub que contiene un ejemplo completo del empleo de la librería DEAP para la solución del problema de la mochila, en la que el alumno puede basarse para proponer su solución al problema planteado. Se puede acceder desde AQUÍ.

La página web del desafío se puede consultar en el siguiente enlace.

Se puede acceder al repositorio de la librería DEAP en el siguiente enlace. Se puede consultar las instrucciones de instalación en la página principal.

La Documentación y Tutoriales de la librería DEAP son accesibles en el siguiente enlace.

Consideraciones de Entrega

La entrega formará parte de la nota de la primera práctica de la asignatura, que se completará en

sesiones posteriores. Se deberá realizar la entrega para poder recibir retro-alimentación del trabajo.

La entrega se realizará en grupos de 1 o 2 alumnos.

Se incluirán en la entrega:

- Fichero fuente en lenguaje python que lee y almacena en memoria los datos.
- El fichero en PDF describiendo la solución propuesta.

Ambos ficheros se comprimirán en un .zip, cuyo nombre seguirá el formato: `ApellidoAlumno1_NombreAlumno1-ApellidoAlumno2_NombreAlumno2.zip`

La entrega se realizará solo por medio del correspondiente espacio de subida en plataforma UBUVirtual. No se admitirán entregas pasado el plazo establecido. En caso de realizarse en pareja, ambos alumnos deberán realizar la entrega.