Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
Centro Académico de Alajuela
Investigación de Operaciones
II Semestre, 2022
Profesor: Carlos Gamboa Venegas

Proyecto 3: Algoritmos Genéticos

Administrativos

- Entrega martes 15 de noviembre. Subiendo en el tec-digital el archivo .zip con el código del repositorio.
- Se deben trabajar en grupos de 2 personas.

Especificación

- Usar Python 3.11 (requerido)
- Debe implementar clases para todo el código con un archivo main.py que cree los objetos y resuelva el problema.
- Debe indicar en un archivo README las bibliotecas de python utilizadas para el correcto funcionamiento del programa,
- Usar archivos de configuración con TOML

Problema de romper contraseña

Se tiene una contraseña de una cantidad de caracteres determinada, los caracteres incluyen dígitos del 0 al 9 y letras de la A a la Z (sin ñ) tanto en mayúsculas como en minúsculas. El programa debe implementar algoritmos genéticos para encontrar la contraseña correcta.

Una solución al problema es una contraseña, esta va a ser el cromosoma. El cálculo de aptitud del cromosoma a ver dato por una función externa llamada calculate_fitness(cromosome), donde el cromosoma es la solución a probar, y esta función devolvera un valor entre 0 y 1 con la valoración de la solución. Esta solución indica que tan buena es, y compara cada unos de los dígitos con la contraseña correcta.

Se proveerá del caparazón para el archivo utils.py, que contiene una clase que implementa dos funciones, init(config) y calculate_fitness(cromosome) donde el cromosoma en un lista con los dígitos de la solución.

En la solución del algoritmo genético debe implementar los tres tipos de selección: ruleta, elite, y ranking. El cruzamiento de los padres utilizando la tasa de crossover, y la probabilidad de mutación de un gene del cromosoma, donde muta a un digito diferente.

En el archivos de configuración implementado con TOML se deberán incluir los siguiente parámetros

El número de padres es la selección que se hace de la población, luego se hace la selección con esos padres que fueron escogidos, claramente se pueden repetir. Se indica que tipo de selección, si es elite se deben mantener la cantidad de mejores padres indicadas en la configuración. El cruzamiento se hace escogiendo. La mutación genera una probabilidad de que un cromosoma cambie un valor de algún gen, seleccionado también de forma aleatoria.

```
[ag]
population_size = 10
num_parents = 5
selection_method = 'ruleta', 'elite', 'ranking'
elite_size = 2
mutation_rate = 0.5
crossover_rate = 90
crossover_method = 'one-point', 'two-point', 'uniform'


[passcode]
correct_passcode = '234AHLp91n'
```

La salida del programa será un despliegue en consola de las generaciones necesarias para crear la solución, y el despliegue de la contraseña encontrada.

Evaluación

Uso correcto del archivo de configuración	10%
Implementación de algoritmo genético	40%
Implementación métodos de selección	15%
Implementación métodos de cruzamiento	15%
Código, documentación interna	10%
Seguimiento de especificaciones	10%
Total	***100%***

Aclaración: si se detecta y confirma la no participación de una persona en el proyecto, esta persona tendrá un 0 en la nota.