SISTEMAS INTELIGENTES I

Bloque 1: Algoritmos Genéticos

Guía de Prácticas

Javier J. Sánchez Medina
javier.sanchez@ulpgc.es

El propósito de esta guía de prácticas es dotar a los/as alumnos/as de Sistemas Inteligentes I, de los rudimentos prácticos en diseño y programación de Algoritmos Genéticos, Redes Neuronales y una introducción a la Minería de Datos

Esto complementa los contenidos teóricos que de esta asignatura serán impartidos.

La asignatura consta de 15 horas de prácticas de Aula y 15 horas de prácticas de Laboratorio. Se dividirán en los tres bloques arriba mencionados:

Bloque 1: Algoritmos Genéticos (6 semanas)
Bloque 2: Redes Neuronales Artificiales (6 semanas)
Bloque 3: Minería de Datos (2 semanas)

Las prácticas son individuales.

Práctica 1: Algoritmos Genéticos. SGA

Introducción:

El objeto de esta práctica es la adquisición de los conocimientos mínimos para con un algoritmo genético sencillo (SGA) resolver un problema de optimización combinatoria no determinística.

Cuando los problemas combinatorios son excesivamente complejos, con espacios de búsqueda inabordables por métodos analíticos ni fuerza bruta, se hace preciso el uso de técnicas de búsqueda "a ciegas". Una de las más populares, con origen en la teoría de la evolución de Charles Darwin, que tiene utilidad en multitud de casos son los Algoritmos Genéticos.

En teoría se explican los fundamentos de los algoritmos genéticos. Podemos resumir en que un algoritmo genético sencillo podría programarse siguiendo el siguiente pseudocódigo

1 – **Inicializar** la población
2 – **Cálculo del Fitness** de cada individuo
3 – **Selección**
4 – **Cruce**
5 – **Mutación**
6 – Mientras **Evaluación de la condición de finalización** sea falso, volver a 2.

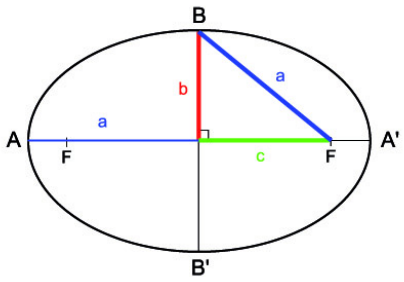
Hay un elemento esencial en el diseño de los algoritmos genéticos y es la codificación del problema. Existen muchos tipos de codificación, y el tipo escogido determina fundamentalmente el diseño del resto de los operadores.

El propósito de esta práctica es el diseño y prueba de un algoritmo genético para un problema sencillo. Partiendo de una imagen binaria de 100 por 100, se trata de encontrar la elipse que mejor se ajuste a la imagen.

La ecuación paramétrica de una elipse, centrada en el origen de coordenadas es:

$$x = a * \cos(\alpha), y = b * \sin(\alpha)$$

Siendo *a* el semieje mayor y *b* el semieje menor.



Si la elipse no está centrada en el eje de coordenadas, la ecuación pasa a ser:

$$x = a * \cos(\alpha) + x_0, y = b * \sin(\alpha) + y_0$$

Si además la elipse está rotada en la imagen, habría que aplicar la ecuación de la rotación en cartesianas, que es:

$$x' = x * \cos(\theta) + y * \sin(\theta),\ y' = x * \sin(\theta) + y * \cos(\theta)$$

Si la aplicamos, a la ecuación de la elipse, el resultado es el siguiente:

$$x' = (a * \cos(\alpha) + x_0) * \cos(\theta) + (b * \sin(\alpha) + y_0) * \sin(\theta)$$

$$y' = (a * \cos(\alpha) + x_0) * \sin(\theta) + (b * \sin(\alpha) + y_0) * \cos(\theta)$$

Objetivos de la práctica:

Los objetivos de esta práctica se pueden resumir como sigue:

1: Diseñar la codificación apropiada para el problema a resolver, utilizando codificación binaria.
2: Diseñar la función de *fitness* que nos permita la detección de las elipses binarias.
3: Diseñar los operadores de selección (truncado + elitismo), cruce (*two points crossover*) y mutación (uniforme), y la condición de finalización (Número fijo de generaciones)
4: Programar un algoritmo genético que nos permita encontrar la elipse/elipses.
5: Ejecutarlo y ajustar los parámetros Tamaño de la Población, Probabilidad de Mutación, número de generaciones (iteraciones), que nos de confianza en encontrar el óptimo en la mayoría de las ejecuciones del algoritmo, evitando la convergencia prematura.
6: Representar la evolución del fitness máximo y medio de la población, y generar una imagen alternativa con la elipse detectada.

Entregables:

Memoria de menos de 10 páginas con un anexo con el código Matlab (u otros) con el algoritmo genético para la detección de elipses en imágenes binarias de 100x100 píxeles.

La memoria debe incluir:

- Codificación utilizada para el cromosoma
- Función de evaluación diseñada
- Operadores de selección, cruce, mutación y condición
- Valores de los parámetros ajustados:
  - Tamaño de la población
  - Número de generaciones
  - Probabilidad de Mutación
- Representación de la evolución del *fitness* máximo y medio para la mejor combinación

Recursos de Ayuda:

NOTAS:

- Tomar como unidad métrica el pixel para la aplicación de las ecuaciones de la elipse.
- Asumir para este ejercicio que todos los cromosomas posibles son individuos “válidos”.

En Matlab:

1: Leer una imagen binaria

En Matlab la instrucción es

```
A = imread(filename)
```

http://es.mathworks.com/help/matlab/ref/imread.html?searchHighlight=imread

2: Si la imagen no es binaria, se puede binarizar con un “threshold” facilmente con la instrucción siguiente:

```
BW = im2bw(I, level)
```

http://es.mathworks.com/help/images/ref/im2bw.html

o manualmente sustituyendo valores por 0 o 1 según estén por encima o debajo de un umbral.

3: Creación de una función en Matlab:

http://es.mathworks.com/help/matlab/ref/function.html?searchHighlight=function

4: Guardar una imagen en Matlab, con ***imsave***:

http://es.mathworks.com/help/images/ref/imsave.html
