Dpto. Sistemas Informáticos y Computación
Escuela Técnica Superior de Ingeniería Informática
UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

Aplicación de metahurísticas poblacionales: Algoritmos Genéticos

Autor: Axel Guzman Godia

Introducción

Este documento tiene como objetivo estudiar el uso de los algoritmos genéticos. Para ello se va a desarrollar un algoritmo genético para solucionar un problema. Además se incluirán detalles de todas sus funciones y un ejemplo de ejecución.

Descripción del problema

Un dúo de músicos deciden pasar un día tocando música en la calle, con el objetivo de conseguir que la gente que pasa les dé algunas monedas. Al final del día, los músicos tienen una bolsa llena de monedas de distinto valor. El problema consiste en repartir las monedas entre los dos músicos, de manera que el reparto sea el más igualitario posible. Para concretar un poco, tenemos una secuencia de monedas de distintos valores (pueden ser repetidos) y hemos de decidir a quién le damos cada moneda.

Representación del problema

Podemos representar las monedas de la bolsa con un vector, donde cada posición es una moneda y contiene el valor de dicha moneda.

Para representar las soluciones a este problema será suficiente un vector de talla N dónde N es el número de monedas. Este vector contendrá solo ceros y unos. Un uno en la primera posición indica que la primera moneda se la queda el primer músico. Un cero indica que se la queda el segundo.

Un ejemplo del problema:

-Tenemos diez monedas, representadas por un vector:

5	1	1	2	10	25	1	1	5	2

-Una posible solución sería:

0	0	0	0	0	0	0	0	0	0

Esta solución indica que todas las monedas se las queda el primer músico. Evidentemente esta solución no es muy buena. Veamos a continuación cómo podemos definir una función "fitness".

Diseño de la función fitness

Para diseñar la función fitness tenemos que tener claro que objetivo perseguimos, es decir , la solución óptima. En este problema en concreto, la solución óptima será aquella que reparta de la manera más justa posible las monedas. Dicho de otra manera, aquella que minimice la diferencia de dinero entre los dos individuos.

Para nuestra función sólo tenemos que tener en cuenta que cuanto menor sea la diferencia entre ambos músicos, mayor será el valor de la función fitness. Una posible función sería la siguiente: **(Atención esta función es sólo ilustrativa)*

$$f(x) = Q - \left(\left| \sum_{i=0}^{N} x_0 * m_i - \sum_{i=0}^{N} x_1 * m_i \right| \right)$$

En esta función, Q representa el valor de todas las monedas (dinero total a repartir). Los sumatorios representan la cantidad de dinero que recibe cada músico**. Nótese que esta representación no es coherente con el vector anterior**, sólo se ha de utilizar como referencia para entender la función fitness. En la aplicación del algoritmo será necesario determinar si el valor del vector solución es 0 o 1 para saber a que miembro se le ha asignado cada moneda. El segundo sumatorio no será necesario porque podemos calcular la cantidad asignada al segundo músico con una simple resta. Aplicando todo esto obtenemos otra versión de la función:

$$f(x) = Q - \left(2 * \left|(Q/2) - \sum_{i=0,}^{N} x_1 * m_i\right|\right)$$

Esta función es básicamente igual que la anterior, pero un poco más sencilla de calcular. Se ha introducido además el término Q/2 para cuantificar como de lejos estamos de un reparto equitativo. Es fácil ver que si el primer músico recibe exactamente la mitad del dinero, el resultado del paréntesis será cero, y obtendremos la fitness máxima. Por el contrario, si un músico se queda con todo el dinero, el resultado del paréntesis será igual a Q; con lo que el fitness total de la función será cero.

A partir de este punto continuaremos explicando el algoritmo sobre una población real.

Aplicación del diseño al problema

Ya tenemos definido cómo vamos a representar nuestras soluciones y cómo vamos a evaluarlas. Todavía nos queda definir diversas funciones, tales como los operadores evolutivos. Sin embargo para hacer la explicación más visual vamos a crear una población inicial de soluciones sobre las que iremos explicando el resto de cosas. Dicho esto también vamos a presentar una instancia concreta del problema:

Tenemos una cantidad total de monedas Q de 67 distribuida en un total de diez monedas. Las monedas que tenemos se expresan en el siguiente vector:

5	1	1	2	10	25	10	1	5	2

Para diseñar nuestra población inicial vamos a construir una población de 4 soluciones aleatorias:

X1									
0	0	0	0	0	0	0	0	0	0
X2									
1	0	0	1	0	0	1	0	0	0
X3									
0	0	0	1	1	1	1	1	1	0
X4									
1	1	1	1	1	1	1	1	1	1

Como ya hemos indicado anteriormente, cada vector expresa a quien vamos a asignar la moneda correspondiente a la posición del vector. Como ejemplo, el vector x4 expresa que todas las monedas se las queda el segundo músico.

Las correspondientes funciones fitness son:

$$f(x_1) = 67 - \left(2 * \left|\left(\frac{67}{2}\right) - 0\right|\right) = 67 - 67 = 0$$

$$f(x_2) = 67 - \left(2 * \left|\left(\frac{67}{2}\right) - (5 + 2 + 10)\right|\right) = 67 - 2 * (16,5) = 34$$

$$f(x_3) = 67 - \left(2 * \left|\left(\frac{67}{2}\right) - (2 + 10 + 25 + 10 + 1 + 5)\right|\right) = 67 - 2 * (19,5) = 28$$

$$f(x_4) = 67 - \left(2 * \left|\left(\frac{67}{2}\right) - 67\right|\right) = 67 - 67 = 0$$

Ahora que tenemos la población inicial vamos a seguir con el proceso. A continuación debemos aplicar los operadores genéticos para crear una nueva población.

Proceso de creación de una nueva población

1.Selección

Para poder generar una nueva población debemos seleccionar aquellos individuos de nuestra población que van a ser padres. Para ello hemos elegido el método de selección por torneo con k=2. Es decir vamos a emparejar parejas de soluciones y a elegir las mejores de estas soluciones. Veamos cómo quedan los emparejamientos y los padres seleccionados:

X1,X4 → X1 X2,X3 → X2

Para cada pareja hemos hecho una selección elitista, es decir hemos escogido el mejor. Ya tenemos nuestros dos padres, ahora tenemos que ver cómo Aplicamos el cruce.

2.Cruce

Para modelar el cruce vamos a aplicar el proceso de cruce uniforme. Esto significa que vamos a seleccionar aleatoriamente partes de los dos padres para generar los nuevos hijos:

Nuestros padres son:

X1									
0	0	0	0	0	0	0	0	0	0
X2									
1	0	0	1	0	0	1	0	0	0

(Para simular la selección aleatoria hemos hecho uso de la página web https://www.random.org/lists/)

Tras aplicar el proceso obtenemos los hijos:

X5									
0	0	0	0	0	0	0	0	0	1
X6									
0	0	0	1	0	0	1	0	1	0

Ahora que ya tenemos el proceso de cruce completado, avanzamos a la siguiente etapa: mutación

3.Mutación

Al escoger el proceso de mutación vamos a tener en cuenta que para que la soluciones del problema sigan siendo coherentes, no debemos permitir métodos que modifiquen los valores del vector fuera de uno o cero. Dicho esto hemos decidido seleccionar el método de mutación por intercambio recíproco. Este método consiste en seleccionar aleatoriamente dos genes e intercambiarlos (Recordemos que cada posición del vector es un gen). La probabilidad de la mutación que hemos escogido es p=0.003. Es probable que en la primera iteración no se produzca ninguna mutación , pero para ejemplificar el uso, vamos a suponer que se produce una mutación en X6. Esta mutación consiste en intercambiar el valor de la posición 3 por el valor de la posición cero. El individuo X6 quedaría:

X6									
1	0	0	**0**	0	0	1	0	1	0

4.Reemplazo

Por último queda aplicar el proceso de remplazo, es decir sustituir los individuos antiguos por los nuevos individuos. Para abordar este paso hemos elegido el método de Gap generacional. Este método consiste en remplazar en cada generación una parte fija de la población. En nuestro caso reemplazaremos un 50% de la población, es decir dos individuos. Para seleccionar que individuos eliminamos, haremos uso de un criterio elitista. Esto significa que los individuos con pero valor de fitness serán reemplazados y aquellos con mayor valor de fitness permanecerán. Una vez aplicado todo el proceso, nuestra población tiene el siguiente aspecto:

X5									
0	0	0	0	0	0	0	0	0	1
X2									
1	0	0	1	0	0	1	0	0	0
X3									
0	0	0	1	1	1	1	1	1	0
X6									
1	0	0	0	0	0	1	0	1	0

Cómo es un problema sencillo, podemos ver que ya se han descartado algunas de las peores soluciones, como son entregar todas las monedas a un solo músico. El siguiente paso en el algoritmo será verificar la soluciones, para saber si tenemos la mejor solución.

Verificación de la solución

En esta etapa del algoritmo vamos a evaluar la soluciones con respecto a una condición de parada. Podemos definir diversas condiciones de parada, como un valor de fitness concreto, mayor que algo, o simplemente un número de iteraciones. Otra opción detener el proceso tras un número determinado de generaciones sin cambios en las soluciones. Para este casp vamos a elegir simplemente un número de iteraciones. Tras unas 30 iteraciones el algoritmo se detendrá. Esto permite obtener una solución rápida, aunque posiblemente no sea la mejor.

Fin del proceso

Hemos mostrado una iteración completa del proceso. A partir de este punto el algoritmo volverá a repetir el proceso de creación de una nueva población, mientras se no se cumpla la condición de parada. Cabe destacar que muchos de los procesos que palica el algoritmo són elegidos por el usuario, y que si eligiéramos procesos diferentes podríamos obtener mejores o peores resultados.