Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

Algoritmos Genéticos (Opt4J)

Nombre: _____

Importante: subid a Poliformat el código generado para cada pregunta (por ejemplo en un .zip).

1. (2 puntos) Debido a una modificación en la normativa de vacunación, ahora se permite que las tres vacunas no se tengan que aplicar necesariamente a la franja de edad de mayores (M1..M5). El problema, por tanto, se mantiene exactamente igual para el caso de jóvenes y adultos, con la misma función multi-objetivo, pero ahora eliminando la restricción sobre la aplicación de las tres vacunas a los mayores.

   Explica el cambio realizado e indica cuál es el conjunto de mejores soluciones encontradas para este nuevo problema tras la ejecución de 800 iteraciones. El resto de los parámetros se deja a libertad del alumno (**NOTA**: indicad claramente cuáles son estos parámetros).

El método de comprobación de restricción ya no se aplica a los mayores:

```
	private Boolean cumpleRestricciones(ArrayList<Integer> fenotipo)
	{
		for (int numVacuna = 0; numVacuna < DatosVacunas.NUM_VACUNAS;
++numVacuna)
		{
			// Jóvenes (J1..J4): 0..3
			// Adultos (A1..A4): 4..7
			// Mayores (M1..M5): 8..12

			if (!vacunaInFranjaEdad(fenotipo, numVacuna, 0, 3) ||
					!vacunaInFranjaEdad(fenotipo, numVacuna, 4, 7)) // ||
//					!vacunaInFranjaEdad(fenotipo, numVacuna, 8, 12))  -->
ya no es aplicable
				return false;
		}

		return true;
	}
```

Hemos relajado restricciones. Se pueden encontrar nuevas soluciones, como por ejemplo (<coste, número de voluntarios>): <80,200>, <82,206>,<87,230>, <89,236>, etc.

   Realiza varias pruebas del algoritmo genético modificando los parámetros “tamaño de la población” y “número de iteraciones”. De acuerdo a las pruebas que has realizado, ¿resulta más adecuado trabajar con una población de mayor tamaño o realizar más iteraciones? Razona la respuesta en base a tus experimentos.

Aunque es algo que no se pueda asegurar, en este problema no he encontrando la solución óptima si se trabaja con un número pequeño de iteraciones (100) aunque el tamaño de la población sea alto (1000). No obstante, en otras ejecuciones podrían obtenerse resultados distintos.

2. (3.5 puntos) **A partir de las modificaciones del ejercicio 1**, se ha detectado que ciertas asignaciones de vacunas a determinados grupos afectan el coste total de aplicación. Concretamente:
   - Si la vacuna3 se aplica a todo el grupo de mayores (M1, M2, M3, M4 y M5) el coste total calculado, según el método inicialmente descrito en la práctica, debe incrementarse en 10 unidades.

   Explica el cambio realizado e indica cuál es el conjunto de mejores soluciones que encuentras para este nuevo problema tras la ejecución de 800 iteraciones. El resto de los parámetros se deja a libertad del alumno (**NOTA**: indicad claramente cuáles son estos parámetros).

NOTA DE IMPLEMENTACIÓN. Recordad que las colecciones en Java comienzan con el índice 0.

Añadimos las siguientes comprobaciones en evaluate

```
// Si la vacuna3 se aplica a todo el grupo de mayores (M1, M2… M5) el coste total se incrementa en 10 unidades
            if ((fenotipo.get(8) == DatosVacunas.NUM_VACUNAS - 1) &&
                        (fenotipo.get(9) == DatosVacunas.NUM_VACUNAS - 1) &&
                        (fenotipo.get(10) == DatosVacunas.NUM_VACUNAS - 1) &&
                        (fenotipo.get(11) == DatosVacunas.NUM_VACUNAS - 1) &&
                        (fenotipo.get(12) == DatosVacunas.NUM_VACUNAS - 1))
                costeTotal += 10;
```

Ahora hemos obtenido estas soluciones de Pareto: <90,238>, <92,244>, <94,246>, <92,244>, etc.

3. (3.5 puntos) **A partir del problema inicial** de las prácticas, deseamos analizar una nueva Vacuna4. El coste de la Vacuna4 es el siguiente:

	Jóvenes				Adultos				Mayores				
	J1	J2	J3	J4	A1	A2	A3	A4	M1	M2	M3	M4	M5
Vacuna4	21	22	21	17	30	29	52	43	33	36	30	34	35

Se debe mantener la restricción original de que "Toda vacuna debe aplicarse a cada franja de edad". Además, esta Vacuna4 debe aplicarse obligatoriamente al grupo M4.

Explica los cambios realizados y cuál es el conjunto de mejores soluciones para este nuevo problema. Todos los parámetros se dejan a libertad del alumno (**NOTA**: indicad claramente cuáles son estos parámetros). Explica qué está ocurriendo en este problema.

Extendemos la matriz de costes:

```
public static final int NUM_VACUNAS    = 4;

{21, 22, 21, 17, 30, 29, 52, 43, 33, 36, 30, 34, 35}      // vacuna4
```

En método evaluate añadimos:

```
// el grupo M4 debe tener la vacuna 4
            if ((fenotipo.get(11) != DatosVacunas.NUM_VACUNAS - 1))
      costeTotal = Double.MAX_VALUE;  // penalizamos el fitness, ya que buscamos minimizar
```

En este caso estamos forzando a que se cumpla una restricción en un grupo determinado (M4). Dado el carácter aleatorio del AG es posible que dicha restricción no se cumpla y no se pueda encontrar una solución factible. De hecho, para conseguir llegar a una solución se han tenido que realizar múltiples pruebas variando los parámetros del algoritmo. Finalmente, se ha conseguido una solución: <213, 291>

Esto pone de manifiesto que un AG es una muy buena opción de optimización, pero en un problema que requiere más etapa de construcción/satisfacción de restricciones AG puede no ser siempre la mejor opción.

4. (1 punto) Explica razonadamente **(no es necesario implementar nada**) cuál sería el mejor genotipo si simplemente se deseara obtener una solución que indicara si a un grupo de voluntarios se le aplica una vacuna distinta del placebo. Es decir, nos da igual saber qué vacuna se ha aplicado y simplemente nos bastaría con saber si es placebo o distinta de placebo.

El mejor genotipo sería `BooleanGenotype`, con 13 valores True/False. Una posible solución sería: "T,T,F,F…T" (con 13 valores).