



Algoritmos Genéticos (Opt4J)

Nombre: Stéphane Díaz-Alejo León

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 la vacuna3 (correspondiente a placebo) no se tenga que aplicar necesariamente a toda franja de edad. El problema, por tanto, se mantiene exactamente igual, con la misma función multi-objetivo, pero ahora eliminando la restricción sobre la vacuna3 y manteniendo las restricciones sobre vacuna1 y vacuna2.

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).

Cambios:

-En mi caso es simplemente eliminar en el Evaluator el boolean que me indicaba si se ponía la vacuna3, es decir, no considerarlo.

Mejores soluciones:

Archive Monitor			
Size: 23		Auto Update <input checked="" type="checkbox"/>	
#	Individual	Coste-MIN (MIN)	Voluntarios-M...
1	[0, 1, 2, 2, 0, 2, 2, 1, 1, 0, 2, 2, 2]	102	278
2	[0, 1, 2, 2, 0, 2, 1, 2, 1, 0, 2, 2, 2]	104	284
3	[0, 1, 1, 2, 0, 2, 2, 1, 1, 0, 2, 2, 2]	109	308
4	[0, 1, 1, 2, 0, 2, 1, 2, 1, 0, 2, 2, 2]	111	314
5	[0, 1, 2, 1, 0, 2, 1, 2, 1, 0, 2, 2, 2]	114	319
6	[0, 1, 1, 1, 0, 2, 2, 1, 1, 0, 2, 2, 2]	119	343
7	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 2, 2]	121	349
8	[0, 1, 1, 2, 0, 2, 1, 1, 1, 0, 2, 2, 2]	126	364
9	[0, 1, 2, 1, 0, 2, 1, 1, 1, 0, 2, 2, 2]	129	369
10	[0, 1, 1, 1, 0, 2, 2, 1, 1, 0, 2, 2, 1]	131	378
11	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 2, 1]	133	384
12	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 2, 2]	136	399
13	[0, 1, 2, 1, 0, 2, 1, 1, 1, 0, 2, 2, 1]	141	404
14	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 1, 1]	144	408
15	[0, 1, 1, 2, 0, 0, 1, 1, 1, 0, 2, 2, 2]	146	412
16	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 1, 2]	147	423
17	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 2, 1]	148	434
18	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 2, 2]	156	447
19	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 1, 1]	159	458
20	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 1, 2]	167	471
21	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 2, 1]	168	482
22	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 1, 1]	179	506
23	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1]	193	526

Parámetros:

-Generations: 800

-Alpha: 100

-Mu: 25

-Lamba: 25

-Crossover Rate: 0.95

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.

Respecto al número de iteraciones se puede percibir que no encuentra ninguna solución óptima mejor después de, más o menos, 100 iteraciones, por lo que aumentar el número de iteraciones no afectaría. En cambio, he podido observar que aumentando el tamaño de la población inicia, produce que tarde más en converger (alcanzar una solución óptima). Con todo esto, razono que es más adecuado trabajar con una población de mayor tamaño, ya que la otra opción, no afecta al caso base.

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 J3 y J4 el coste total calculado, según el método inicialmente descrito en la práctica, debe reducirse en 5 unidades.
 - Si la vacuna3 se aplica a A2 y A3 el coste total debe incrementarse en 4 unidades.

Explica el cambio requerido 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).

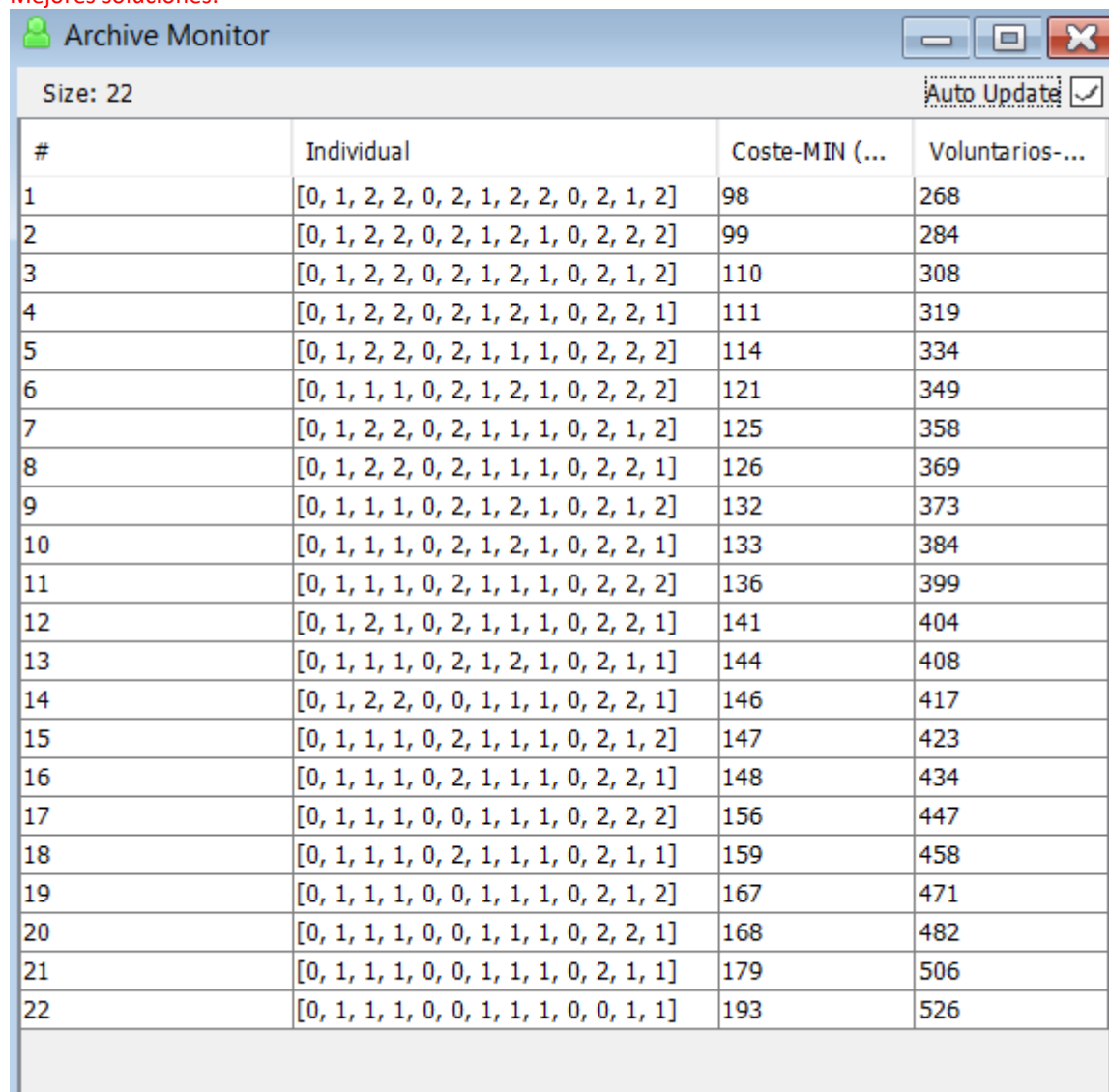
NOTA DE IMPLEMENTACIÓN. Recordad que las colecciones en Java comienzan con el índice 0.

Cambios:

-En mi caso es añadir en el Evaluador dos if que tenga en cuenta las condiciones y resten o aumenten el coste total.

```
if(phenotype.get(2) == 2 && phenotype.get(3) == 2) {  
    coste -= 5;  
}  
if(phenotype.get(5) == 2 && phenotype.get(6) == 2) {  
    coste += 4;  
}
```

Mejores soluciones:



Archive Monitor

Size: 22 Auto Update ☒

#	Individual	Coste-MIN (...)	Voluntarios-...
1	[0, 1, 2, 2, 0, 2, 1, 2, 2, 0, 2, 1, 2]	98	268
2	[0, 1, 2, 2, 0, 2, 1, 2, 1, 0, 2, 2, 2]	99	284
3	[0, 1, 2, 2, 0, 2, 1, 2, 1, 0, 2, 1, 2]	110	308
4	[0, 1, 2, 2, 0, 2, 1, 2, 1, 0, 2, 2, 1]	111	319
5	[0, 1, 2, 2, 0, 2, 1, 1, 1, 0, 2, 2, 2]	114	334
6	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 2, 2]	121	349
7	[0, 1, 2, 2, 0, 2, 1, 1, 1, 0, 2, 1, 2]	125	358
8	[0, 1, 2, 2, 0, 2, 1, 1, 1, 0, 2, 2, 1]	126	369
9	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 1, 2]	132	373
10	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 2, 1]	133	384
11	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 2, 2]	136	399
12	[0, 1, 2, 1, 0, 2, 1, 1, 1, 0, 2, 2, 1]	141	404
13	[0, 1, 1, 1, 0, 2, 1, 2, 1, 0, 2, 1, 1]	144	408
14	[0, 1, 2, 2, 0, 0, 1, 1, 1, 0, 2, 2, 1]	146	417
15	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 1, 2]	147	423
16	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 2, 1]	148	434
17	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 2, 2]	156	447
18	[0, 1, 1, 1, 0, 2, 1, 1, 1, 0, 2, 1, 1]	159	458
19	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 1, 2]	167	471
20	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 2, 1]	168	482
21	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 2, 1, 1]	179	506
22	[0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1]	193	526

Parámetros:

-Generations: 800
-Alpha: 100
-Mu: 25
-Lamba: 25
-Crossover Rate: 0.95

3. (3.5 puntos) **A partir del problema inicial** de la práctica, 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	20	21	16	20	35	28	50	40	31	34	31	32	31

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 J4.

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.

Cambios:

-Modificación del archivo con los datos.
-En el evaluador se ha añadido el boolean para tener en cuenta la condición de que se ponga la vacuna 4
-En el evaluador se ha añadido un if que invalida si no se ha puesto la vacuna4 al grupo J4

```
if (!(phenotype.get(3) == 3)) {  
    inv = true;  
}
```

Mejores soluciones:

Archive Monitor			
Size: 5		Auto Update <input checked="" type="checkbox"/>	
#	Individual	Coste-MIN (MIN)	Voluntarios-MAX ...
1	[0, 1, 2, 3, 0, 3, 2, 1, 2, 0, 2, 1, 3]	173	262
2	[0, 1, 2, 3, 0, 3, 2, 1, 1, 0, 2, 2, 3]	174	278
3	[0, 1, 2, 3, 0, 3, 1, 2, 1, 0, 2, 2, 3]	176	284
4	[0, 1, 2, 3, 0, 3, 2, 1, 1, 0, 2, 1, 3]	185	302
5	[0, 1, 2, 3, 0, 3, 2, 1, 1, 0, 2, 3, 1]	186	313

Explicar problema:

Se puede observar que converge rápido y que se encuentran pocas soluciones óptimas.

Parámetros:

-Generations: 800
-Alpha: 100
-Mu: 25
-Lamba: 25
-Crossover Rate: 0.95

4. (1 punto) Explica razonadamente (**no es necesario implementar nada**) cuál sería el mejor genotipo si se desearan probar 13 vacunas distintas, una por grupo de voluntarios, sin repetir ninguna. Es decir, cada grupo solo recibe una vacuna y cada vacuna se aplica a un solo grupo.

El mejor genotipo sería `PermutationGenotype<Integer>` ya que generaría una lista de valores enteros en los que no hay repeticiones, por lo que asignaría una vacuna distinta a cada grupo.