Práctico 1

Gonzalo Torterolo
Facultad de ingeniería
UDELAR
Montevideo, Uruguay
Email: gonzalo.torterolo@fing.edu.uy

Gisel Cincunegui
Facultad de ingeniería
UDELAR
Montevideo, Uruguay
Email: gisel.cincunegui@fing.edu.uy

***Abstract*—En el presente informe el objetivo es probar las ventajas y limitaciones de las actuales librerias que implementan algoritmos genéticos mediante la resolución de 2 problemas típicos de computación. Además se realizan pequeos analisis de las soluciones con el fin de verter conocimientos teóricos adquiridos en el curso.**

***Index Terms*—Algoritmos Evolutivos, AE, Algoritmos Genéticos, AG, Malva, Mallba, Whatchmaker, Knapshak, Mochila**

I. Introducción

En la consigna presentada (ver letra en [1]) se deben resolver dos variantes del problema de la mochila (explicadas en la consigna también) utilizando ideas de los algoritmos genéticos.

Por otra parte, el análisis y modelado del problema así como las ventajas de utilizar este tipo de algorimos para el escenario queda relegado a una segunda instancia, pues ya estaban resueltos en las consignas. En todo momento primó la aplicación de los conceptos teóricos del curso antes que la obtención de resultados como rendimiento, modularidad, etc. No nos interesa, por lo tanto, seguir aquí un enfoque práctico para esta primer aproximación a los algoritmos evolutivos donde ambos problemas son típicos y ampliamente estudiados.

A. Aplicación de AG

Se analizan las diferentes soluciones a las dos variantes del problema de la mochila aquí presentadas, concluyendo con una implementación para ambos casos. Para comenzar a definir las soluciones a los problemas se debe determinar las siguientes carácterízticas de importancia para un AE típico.

1) Codificación.
   - Tratamiento de codificaciones no factibles.
2) Población inicial.
   - Evaluación de coste/beneficio de utilizar criterios inteligentes de inicialización.
   - Velocidad de convergencia a un óptimo.
3) Funcion de fitness.
   - Carácterízticas especiales requeridas (e.g.: no negatividad de la función para seleccíon proporcional)
   - Transformación para otros escenarios (e.g.: aplicación para maximización/minimización)
   - Ajustes para mejorar soluciones (escalado, parámetros de configuración).
4) Seleccción:
   - Elitismo o no.
5) Cruzamiento.
6) Mutación.
7) Remplazo.

B. Particularidades y observaciones de las implementación/Librerias

- Malva
  - Malva utiliza otro concepto para la mutación, la probabilidad se aplica por individuo y no por gen.
  - Parece bastante desprolija, aunque quizás mucho más performante y escalable que cualquier otra, pero como ya se dijo, estas carácterízticas no interesan en este momento.
- Whatchmaker
  - Permite elitismo y esta cantidad es extra a la de la población especificada.

C. Problemas de los aperitivos

Para este primer problema interesa obtener el óptimo de un problema. La aplicación de AE no nos parecío muy adecuada, pues no se estaba buscando una solucin aceptable, sino que solo la mejor solucin posible es aceptable para los comensales. Por otra parte, el problema es multimodal y NP computacionalmente dificil, lo que lo hace atractivo a una solucin de AG. En esta variante del problema de la mochila se buscan soluciones enteras pero se aceptan repeticiones, y no existen pesos, o puede entenderse que el peso es proporcional a la ganancia o costo (es el mismo). Se utiliz la libera Malva en este problema. Se definen las caracterizticas de la solucin relativas al AG:

Notacin

$$X_i, C_i, O$$

son la cantidad de aperitivos, costo de aperitivo para el tipo i y costo objetivo

1) Codificación. Nos es importante utilizar una codificacin conveniente a para facilitar el trabajo del cruzamiento y utilizar SPX o cruzamiento uniforme que ya se encuentran implementadas.
2) Evaluamos la posibilidad de utilizar 2 tipos de codificaciones. La primera de ella muy parecida a la pedida en el prximo problema, donde asignamos un gen binario (los alelos toman valores en $\{0,1\}$) para cada posible aperitivo en caso de que se elija o no. En el caso deberamos asignar una cota a la cantidad del genoma, dada por **??** para cada tipo de aperitivo. Esta codificacin es bastante intuitiva pero tiene la desventaja de asignar varios genotipos a un mismo fenotipo. El cruzamiento SPX tiene un comportamiento menos discruptivo en esta representacin. Podra ser til en etapas avanzadas de la evolucin.

   $$M_i = \lceil \frac{N}{C_i} \rceil \tag{1}$$

   Otra codificacin elimina el problema de que varios genotipos se correspondan con un mismo fenotipo. La misma consite en una representacin de enteros, donde cada gen representa la cantidad

de aperitivos de un tipo dado. Aplicada junto a un cruzamiento SPX esta codificacin tiene un comportamiento ms discruptivo que la anterior, se mueve por bloques.

Se puede incluso, utilizar ambas representaciones en distintas etapas de la evoluciones, de hecho propondremos una funcin de fitness que es compatible para ambas codificaciones.

En las codificaciones anteriores se le puede agregar semntica en el ordenamiento del cromosoma. Se discute la posibilidad de ordenar los genes dentro del cromosoma en funcin del costo. Para la implementacin se elige la segunda opcin, sin agregar informacin en el ordenamiento.

Todas las soluciones representables sern consideraciones factibles.

3) Seleccin Usaremos seleccin proporcional tanto para la seleccin de cruzamiento como seleccin generacional, ya implementada en malva utilizando el algoritmos de la ruleta. La funcin de fitness toma valores positivos, por lo que es adecuada para este tipo de seleccin. proporcin de seleccin pj = Fj/Sum(Fi)
4) Fitness Tomaremos como funcin de fitness la distancia al valor objetivo. Podemos configurar la librera Malva para resolver un problema de minimizacin en vez de maximizacin, por lo que no hac falta transformar la funcin de fitness.

$$v = |\sum_{i=0}^{N}(C_i X_i) - O|$$

es la distancia del valor objetivo.

5) Cruzamiento Se ha enfocado la eleccin de las anteriores caracterzticas de AG para utilizar SPX, pero otros algoritmos de cruzamientos donde se apliquen operadores numricos podra surgir, como por ejemplo, se propone el promedio entre valores de cada padre, o diferentes promedios ponderados. Sin embargo, para la implementacin se opta por SPX.
6) Mutacin Se utiliz una mutacin uniforme sobre los valores posibles de alelo de cada gen, expresada por la formula:

$$B_i = (B_i + rand(0, M_i))\%M_i$$

Poblacin inicial La poblacin inicial ser muy importante para determinar una buena solucin, pues a partir de ella se tiene casi todo el material gentico nuevo. En las ejecuciones donde se ha probado la probabilidad de mutacin baja hace que no se genere nuevo material gentico ms del aportado por la poblacin inicial. Sin embargo, una cantidad suficiente de individuos en cada generacin logra cubrir el espacio de busqueda bastante bien. Para la generacin de la poblacin inicial se utiliz un criterio aleatorio, pero se asegura que existen al menos un aperitivo de cada tipo al asignar el mximo valor factible a en algn gen de cada individuo (el gen numero de individuo % cantidad de genes del cromosoma).

D. Problemas de la mochila

1) Descripcion del problema: El problema es un problema de optimizacin combinatoria perteneciente a la familia de problemas N-P difciles. El problema se define de la siguiente manera: Dados un conjunto de n objetos, cada uno con una ganancia asociada gi y un peso asociado pi, el objetivo del problema es encontrar el subconjunto de objetos que maximiza la ganancia total, manteniendo el peso total por debajo de la capacidad mxima de la mochila (W).

$$x = \sum_{i=0}^{n} g_i x_i \tag{2}$$

$$x = \sum_{i=0}^{n} p_i x_i < W \tag{3}$$

Descripcin de la solucin: Este problema constiste en obtener una solucin lo mas prxima posible a la ptima del problema, implementando para ello tcnicas de algoritmos evolutivos.

A continuacion se detallar las desiciones para la implementacion del algoritmo evolutivo del problema de la mochila:

Codificacion: Arreglo binario con el largo igual a la cantidad de elementos totales que puedo incorporar en la mochila, donde cada posicion del arreglo correspone a el objeto i de la solucin. Dicho arreglo ser ordenado por peso, manteniendo la referencia del ordenamiento del arreglo original, ya que se busca con esto que la operacion de cruzamiento operada por el algoritmo evolutivo intente probar distintos valores de ganancia "manteniendo" relativamente el peso de la solucion mas factible. El odenamiento de el arreglo se har manteniendo dos estructuras de indices para la codificacion (arreglo original -¿ arreglo ordenado , fenotipo -¿ genotipo) y la decodificacion (arreglo ordenado -¿ arreglo original, genotipo -¿ fenotipo). La funcin de fitness se evaluar con el fenotipo, las operaciones se realizaran con el genotipo, y el resultado retornar el fenotipo de la mejor solucion encontrada.

Factibilidad de la solucion: Se tomar como solucion factible aquella que no sobrepase el limite de peso de la mochila.

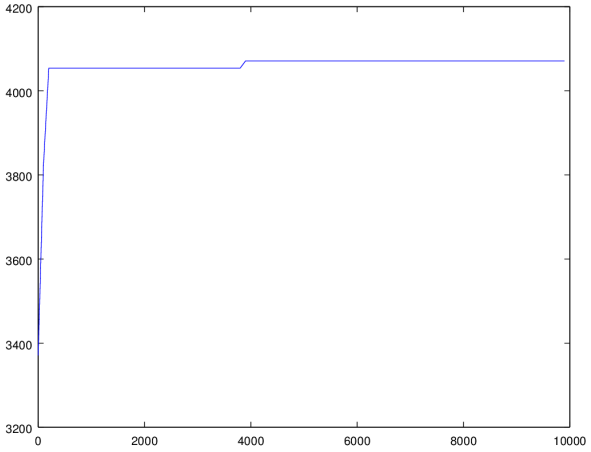
Poblacin inicial: Para la poblacin inicial se opt por la generacion aleatoria de los individuos.

Funcion de fitness: La ecuacion final de la misma ha mutado de acuerdo a proble 5. Cruzamiento. SPX. Idema anterior.mas que se nos han planteado a lo largo de la implementacion. Al principio se opt por lo siguiente: -Funcion con penalizacion: La idea de esta funcin es mantener los individuos no factibles a lo largo de las generaciones para evitar la deriva gentica, ya que los mimos pueden brindar informacion relevante para llegar a la optimizacion de la solucion final. Si el peso total del individo no sobrepasa al maxmo permitido para la mochila(candidato factible), la funcin de fitness retornar la ganancia total del individo evaluado. Si el peso sobrepasa el mximo estipulado (candidato no factible), la ganancia total se penalizar restando la ganancia proporcional a la diferencia de peso, quedando a formla como la siguiente: Gtot-((2*Gtot*(k-p))/k) -Funcion sin penalizacion: Como la solucin anterior present problemas con respecto a la solucion retornada por el algoritmo, debido a que se devolvian soluciones no factibles, se opt por solo maximizar la ganancia, e implementar otros mecanimos para cuando nos topamos con casos de no factibilidad que se detallar ms adelante.

Seleccin: Para la seleccin se opt utilizar el mecanismo de la seleccin proporcional implementada con el algoritmo de la ruleta, para poder mantener individuos menos factibles en la poblacion generacional y as evitar la deriva genetica.

Cruzamiento SPX. Idema anterior. Cruzamiento con probabilidad 0.75 Para evitar el problema de obtener individuos no factibles, lo que se realiz fue que luego de aplicar la operacion de cruzamiento se evala la factibilidad de los mismos, si estos no son factibles se realizar una mutacin a el objeto que tenga menos peso, con valor 1 en la codificacion. (otra opcion seria mutar algun objeto aleatoriamente o a aquel que posea menos ganacia), y aplicar esto hasta conseguir la factibilidad del candidato. Asi, logramos no tener candidatos no factibles, y

por lo tanto no tener soluciones no factibles.
Mutacin. Mutacin de bit aleatorio con probabilidad 0.01.
Condicin de parada: 10000 generaciones.



$$Z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + a + b \tag{4}$$

$$Z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + a + b \tag{5}$$

$$A_1 = 7 \tag{6a}$$
$$A_2 = b + 1 \tag{6b}$$

and

$$A_3 = d + 2 \tag{6c}$$

$$|x| = \begin{cases} x, & \text{for } x \geq 0 \quad (7a) \\ -x, & \text{for } x < 0 \quad (7b) \end{cases}$$

$$Z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + a + b \tag{8}$$
$$+a + b \tag{9}$$
$$+ a + b \tag{10}$$
$$+ a + b \tag{11}$$

II. Conclusión

Al final de este laboratorio hemos podido decicir aquella librería que se adapta mejor a nuestras necesidades y que nos gustaría usar para resolver el proyecto final. Además, en el proceso de realización de este primer práctico, logramos interiorizarnos en aspectos prácticos y la aplicación de algoritmos genéticos y la síntesis de informes con Latex y otras herramientas.

References

[1]