Práctica 3

Metaheurísticas

Algoritmos Genéticos para el problema del Clustering

Grado en Ingeniería Informática

Alumno: José Arcos Aneas
DNI: 74740565-H
Grupo: *Lunes 14h-16h*

Índice

0. Introducción

Se presentan dos algoritmos genéticos diferentes, uno generacional elitista, y otro estacionario, además se presenta el algoritmo K-MM como algoritmo de comparación.

El código a sido generado en C. Se adjunta una carpeta llamada Códigos con unos archivos *.c* y *.h* junto con los archivos de configuración y los archivos .txt para la entrada de datos. Además contiene un archivo makefile para compilación de los distintos archivos para generar el ejecutable.

Ahora explicare como esta estructurada esta práctica.
El punto uno contiene una breve descripción del problema similar a la de la práctica anterior. A continuación describo los algoritmos de los que consta esta entrega ademas de algunos operadores importantes en estos algoritmos.
Los siguientes dos puntos (3,4), describo a cada uno de los algoritmos utilizados con mayor profundidad introduciéndome mas en la implementación.
El punto siguiente esta dedicado a la comparación de los resultados obtenidos de las diferentes ejecuciones.

Por ultimo también se adjunta en la entrega unas tablas con los resultados obtenidos en las diferentes ejecuciones de cada uno de los algoritmos y sobre cada uno de los ficheros de entrada que se proporcionaban en la web de la asignatura, aunque se probaron además en mas ficheros de entrada y con diferentes implementaciones de la búsqueda local, siendo los resultados que se presentan los que mejor mas acertados me parecieron.

1. Descripción del problema.

El problema que abordamos es el del clustering. El clustering puede definirse como el proceso de agruupar un conjunto de objetos físicos o abstractos dentro de clases con objetos similares se denomina clustering. El clustering consiste en agrupar una colección dada de datos no etiquetados en un conjunto de grupos de tal manera que los objetos que pertenecen a un grupo sean homogéneos entre sí y buscando además, que la heterogeneidad entre los distintos grupos la sea lo más elevada posible . Podríamos ver un problema de clustering como un problema de optimizaron que localiza los centroides óptimos de los clusters en lugar de encontrar la partición del espacio, cayendo en óptimos locales.

El problema que abordamos es el del clustering. El clustering puede definirse como el proceso de agrupar un conjunto de objetos físicos o abstractos dentro de clases con objetos similares se denomina clustering. El clustering consiste en agrupar una colección dada de datos no etiquetados en un conjunto de grupos de tal manera que los objetos que pertenecen a un grupo sean homogéneos entre sí y buscando además, que la heterogeneidad entre los distintos grupos la sea lo más elevada posible . Podríamos ver un problema de clustering como un problema de optimizaron que localiza los centroides óptimos de los clusters en lugar de encontrar la partición del espacio, cayendo en óptimos locales.

En la configuración de los clúster cada objeto permanece a un único clúster, y su distancia

al centroide de éste es siempre menor que a los restantes. Así, suponiendo un conjunto M formado por m objetos y configuración de k clústeres C_i, debe cumplirse:

a) $C_i \neq \emptyset$, es decir, $|C_i| > 0$.

b) $C_i \cap C_j = \emptyset$, para todo $i \neq j$.

c) $\sum_{i=1}^{k} C_i = M$, donde:

- M = {X^1, …, X^m} es el conjunto de patrones
- X^i = $(x_1^i, \ldots, x_n^i)$, con $x_j^i \in R$.

En el algoritmo que implementaré intentaré deducir J, según la siguiente formula:

$$Minimizar\ J = f(W, Z; X) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_{ij} \| X^i - Z^j \|^2$$

Sujeto a que:

- $\sum_{j=1}^{k} w_{ij} = 1, 1 \leq i \leq m$
- $w_{ij} \in \{0,1\}, 1 \leq i \leq m, 1 \leq j \leq k$

Donde:

- X^i = $(x_1^i, \ldots, x_n^i)$, con $x_l^i \in R$ es el i-ésimo patrón de M.
- Z^j = $(z_1^j, \ldots, z_n^j)$, con $z_l^j \in R$ es el centroide del clúster C_j.
- W es una matriz de pertenencia con dimensión mxk en la que, un dato pertenece a un solo centroide, representando este pertenece con un 1, y un 0 en el caso contrario.
- K es el número de clústeres, m es el número de patrones en M, n es la dimensión del espacio y ||.|| es la distancia Euclídea al cuadrado.

Los algoritmos que presentamos para resolver este problema son un algoritmo genético generacional elitista y otro estacionario con las consideraciones que se remarcan más adelante.

2. Descripción de los algoritmos

Este problema será representado por un conjunto de parámetros (genes), éstos pueden ser unidos para formar una cadena de valores (cromosoma), a este proceso se le llama codificación.

Algoritmos genéticos generacionales

En tal modelo se realizan cruces en una población de individuos, los descendientes son colocados en otra. Al final de la fase reproductiva se elimina la generación anterior y se utiliza la nueva. Este modelo también es conocido como algoritmo genético canónico.

Cruce

El cruce se lleva a cabo sobre el conjunto intermedio generado por la reproducción. Primero se selecciona aleatoriamente una pareja de individuos para ser cruzados. Después, con el uso de la teoría de las probabilidades se determina si habrá cruce entre los dos individuos seleccionados o no.

No todas las parejas de cromosomas se cruzarán, sino que habrá algunas que pasarán intactas a la siguiente generación. Usaremos una técnica llamada elitismo, en la que el individuo más apto a lo largo de las distintas generaciones no se cruza con nadie y se mantiene intacto hasta que surge otro mejor que lo desplazará.

Mutación

La mutación es aplicada a cada descendiente individualmente después de cada cruce. Esta altera cada uno de los genes del cromosoma al azar, con una probabilidad pequeña. . Este operador permite la introducción del nuevo material cromosómico en la población, tal y como sucede con sus equivalentes biológicos. Al igual que el cruce, la mutación se maneja como un porcentaje que indica con qué frecuencia se efectuará, aunque se distingue de la primera por ocurrir mucho más esporádicamente. Este bajo por ciento en la probabilidad de mutación evita oscilaciones en el promedio de los valores objetivos de la población. Implementamos la mutación probabilidad fija. La mutación busca mejorar aún más las soluciones creadas con los dos operadores anteriores. Se puede decir entonces que los algoritmos genéticos utilizan soluciones que por su evaluación han demostrado ser buenas para combinarlas y producir otras todavía mejores.

Algoritmos genéticos estacionarios.

Si aplicamos la estrategia de reemplazar al peor individuo podriamos coducir a una convergencia prematura.

• La estrategia de reemplazo aleatorio funciona correctamente en general, aunque la convergencia es lenta.
• Para disminuir los efectos adversos puede utilizarse reemplazo probabilístico, inversamente proporcional a los valores de fitness.

La mayor diferencia entre el modelo de estado estacionario y el modelo generacional consiste en que para cada P miembros de la población creados, el modelo de estado estacionario necesita realizar 2 selecciones a través de un torneo binario.

• Como consecuencia, la presión de selección y la “deriva genética” (pérdida de información) será el doble en un algoritmo evolutivo de estado estacionario.
• El algoritmo de estado estacionario es más veloz para encontrar buenas soluciones que el algoritmo generacional, aunque si el plazo fuera mas largo ,(si la diferencias entre las ejecuciones fuera muy grande) puede que sus soluciones sean superadas por que el algoritmo generacional, tiene un mejor patrón de exploración del espacio de búsqueda.

El ciclo de vida y muerte de individuos se relaciona con el manejo de la población (selección, reemplazo), los individuos tienen un tiempo de vida asociado.

• En general, el tiempo de vida esperado de un individuo es de una generación, sin embargo, en algunos AGE puede ser mayor.
• Las estrategias de elitismo relacionan el tiempo de vida de un individuo con su valor de fitness. Son técnicas para mantener las buenas soluciones por un tiempo superior a una generación. El mecanismo evolutivo converge cuando los individuos de la población son idénticos o muy parecidos entre si. El operador de cruce deja de producir nuevos individuos y la búsqueda se estanca.
• Cuando este comportamiento se da antes de encontrar la solución óptima de un problema, el AE sufre de convergencia prematura.
• El operador de mutación provee un mecanismo para reintroducir patrones perdidos, pero es poco efectivo (ya que es probabilístico y muy lento).
• Se han propuesto varios métodos para reducir la probabilidad de pérdida de información genética y mantener la diversidad.

Siendo G la población en un tiempo t podríamos describir el algoritmo genético como.

```
generar población inicial, G(0);
evaluar G (0);
t:= 0;
repetir
        t:= t + 1
        generar G (t) usando
        G (t-1);
        evaluar  G (t);
hasta encontrar una solución;
```

Cada iteración de este bucle es conocida como generación. La primera de este proceso es una población de individuos generados al azar. Desde ese punto, los operadores genéticos, unidos a la medida de adaptación, actúan para mejorar la población.

3. Descripción Algoritmo Genético Generacional Elitista.

Esquema de representación.

Función objetivo. Función double J()

```
j(){
        recorrer todos los datos
        calcular la suma de las distancias Patrón al centroinde.
}
```

Generacional de la población inicial.

```
        Inicializar()
```

Esquema de evolución: se seleccionará una población de padres del mismo tamaño que la población genética.

```
SeleccionAGG()
        for(0,tamaño de la poblacion,1)
                elegir dos
                torneo binario
        fin del for
fin seleccionAGG
```

Además aquí se aplica el torneo binario tantas veces como el tamaño de la población, incluyendo en este método el operador de selección.

Esquema de reemplazamiento: En el esquema generacional, la población de hijos sustituye automáticamente a la actual. Para conservar el elitismo, si la mejor solución de la generación anterior no sobrevive, sustituye directamente la peor solución de la nueva población.

```
ReemplazarAGG()
        Buscamos la mejor solución de nuestra población.
        Comprobamos si la población a cambiado.
                si a cambiado y la peor solución es menor
                        sutituimo el peor de la nueva por el mejor de nueva.

fin ReemplazarAGG
```

Operador de cruce: Se empleará el operador de cruce OX para representación de orden, explicado en las transparencias de teoría del tema de Algoritmos Genéticos.

Operador de mutación: Se considerará el operador de intercambio para representación de orden, explicado en las transparencias del Seminario 3.a. Para aplicarlo, se escogerá aleatoriamente otro gen con el que intercambiar el patrón existente en el gen a mutar.
El tamaño de la población será de 50 cromosomas. La probabilidad de cruce será 0,7 en el AGG.La probabilidad de mutación (por gen) será de 0,01 en ambos casos. El criterio de parada en las dos versiones del AG consistirá en realizar 20000 evaluaciones de la función objetivo. Se realizarán cinco ejecuciones sobre cada caso del problema

```
MutacionAGG()
        Generamos población a mutar
                elegimos un cromosoma
                elegimos dos genes
                sustituimos uno por otro
                marcamos cromosoma como cambiado.
fin MutacionAGG()
```

4- Descripción Algoritmo Genético Estacionario.

Esquema de representación.

Función objetivo. Función double J()

```
        j() {
        recorrer todos los datos
        calcular la suma de las distancias Patron al centroinde.
        }
```

Generacional de la población inicial.

```
         Inicializar(){

                recoremos los datos y los metemos en las estructuras
                luego le damos valores aleatorios
                posteriormente evaluamos
                }
```

Esquema de evolución: se seleccionará una población de padres del mismo tamaño que la población genética.

SeleccionAGE()
    for(0,2,1)
        elegir dos
        torneo binario
    fin del for
fin seleccionAGe

Además aquí se aplica el torneo binario tantas veces como el tamaño de la población, incluyendo en este método el operador de selección.

Esquema de reemplazamiento: En el esquema generacional, la población de hijos sustituye automáticamente a la actual. Para conservar el elitismo, si la mejor solución de la generación anterior no sobrevive, sustituye directamente la peor solución de la nueva población.

ReemplasarAGE()
    Buscamos las dos peores solución de nuestra población.
    Comprobamos si la población a cambiado.
        si a cambiado y la peor solución es menor
            sustituimos el peor de la nueva por el mejor de nueva.

fin reemplazar.

Operador de cruce: Se empleará el operador de cruce OX para representación de orden, explicado en las transparencias de teoría del tema de Algoritmos Genéticos.

Operador de mutación: Se considerará el operador de intercambio para representación de orden, explicado en las transparencias del Seminario 3.a. Para aplicarlo, se escogerá aleatoriamente otro gen con el que intercambiar el patrón existente en el gen a mutar.
El tamaño de la población será de 50 cromosomas. La probabilidad de cruce será 0,7 en el AGG. La probabilidad de mutación (por gen) será de 0,01 en ambos casos. El criterio de parada en las dos versiones del AG consistirá en realizar 20000 evaluaciones de la función objetivo. Se realizarán cinco ejecuciones sobre cada caso del problema

MutacionAGe()
    Generamos prblacion a mutar
        elegimos un cromosoma
        elegimos dos genes
        sustituimos uno por otro
        marcamos cromosoma como cambiado.
fin MutacionAGe()

5. Comparación con algoritmo K- medias.

En este apartado procedo a compara los dos algoritmos anteriores con los resultados obtenidos al ejecutar el k-mm.

En general los algoritmos genéticos son mas rápidos y obtienen mejores resultados que el k-mm. Ya que requiere un numero de ejecuciones elevado pero al compararlo con el k-medias. Vemos que aunque el k-medias puede tardar sobre 1-2 segundos en ejecutarse (en algunos casos), los resultados son en todos los casos peores que los que podemos observar en los genéticos en los que el máximo tiempo de ejecución puede ser de 3 min en el archivo mas grande (aggregation) mientras que el k medias 1 min.

- **AGG vs K-medias**

En general para todos los archivos ejecutados los tiempos de ejecución de un algoritmo k-medias mejora los resultados obtenidos al ejecutar los algoritmos genéticos generacionales elitistas.
Gracias al evaluar también la técnica de multiarranque podemos observar que aunque ejecutemos mas veces el k-medias no sera capa de mejorar los tiempos de y los resultados de AGG.

- **AGE vs K-medias**

La mejora de en este caso en los resultados es menos evidente que en el caso anterior, pero la diferencia en tiempo no es tanta, casi igualando a la del k-medias.

El uso de algoritmos genéticos mejora los resultados del k-medias.
La mejora mas drástica en tiempo la presentan los algoritmos genéticos estacionarios.

En algunos casos, para archivos pequeños como aggregation.txt los tiempo de los algoritmos genéticos se acercan mucho a k-medias, y en todos los casos mejoran los resultados obtenidos en cualquier ejecución del k-medias en valor mínimo de J.

Entre los algoritmos aplicados AGG y AGE podemos concluir que como ya habíamos dicho antes, los tiempo de ejecución del algoritmo estacionario es mucho mejor pero puede que sus soluciones sean superadas por que el algoritmo generacional, tiene un mejor patrón de exploración del espacio de búsqueda.