Centro de Investigación en Matemáticas (CIMAT).
Unidad Monterrey

Algoritmo Memetico: Problema P-Mediana

Marcelo A. Sanches Zaragoza
Karla M. Reyes Maya

26 de noviembre de 2021

1. Problema P-Mediana

1.1. Parámetros

- I : conjunto de instalaciones
- J : conjunto de clientes
- C_{ij} : costo de asignación del cliente j a la instalación i, $\forall i \in I$, $j \in J$
- f_i costo de localizar la instalación i, $\forall i \in I$
- p : número de instalaciones que se deben abrir

$$x_{ij} = \begin{cases} 1 \textit{ si el cliente } j \textit{ se asigna a la instalacion } i \\ 0 \textit{ otro caso} \end{cases}$$

$$y_i = \begin{cases} 1 \textit{ si la instalacion abre} \\ 0 \textit{ otro caso} \end{cases}$$

1.2. Modelo

Minimizar:

$$z = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{i \in I} f_i y_i$$

Sujeto a:

$$\sum_{i \in I} y_i = p$$

$$\sum_{j \in J} x_{ij} = 1 \ \forall i \in J$$

$$x_{ij} \leq y_i \ \forall i \in I, j \in J$$

$$y_i \in \{0, 1\}, x_{ij} \in 0, 1 \ \forall i \in I, j \in J$$

1.3. Restricciones

En la ecuación (1) la función objetivo es minimizar el costo de asignación y el costo de apertura, la restricción (2) asegura que sólo se abra la cantidad de instalaciones indicada, la restricción (3) garantiza que cada cliente sea asignado a una sola instalación, la restricción (4) asegura que cada cliente sea asignado solo a una instalación abierta y finalmente la restricción (5) es la restricción de signo de las variables de decisión.

2. Pseudocodigo

Algorithm 1 Algoritmo Genético para problema P-Mediana

Require: I, J, C_{ij}, f_i, p, N_gen, K, $prob_{cruza}$, $Tolerancia$

Generación Población Aleatoria
$P's \longleftarrow random[P_1, \cdots, P_p]$ tomada del conjunto I con cardinalidad p.
$P_0 \longleftarrow binario[P's]$ codificación binaria de $P's$ para las instalaciones
for 0 hasta N_gen **do**
  function Torneo de padres(P_0,$P's$,K)
    Seleccionar una vecindad de K individuos aleatoriamente
    Evaluar $Fitness(P's)$ de los K individuos
    Seleccionar 2 mejores Fitness
    Seleccionar los P_0 con los mejores $Fitness(P's)$
    return $Padres\_P\_0 \longleftarrow$ 2 Cromosomas P_0 con los mejores Fitness
  end function
  function Generacion Hijos(P_0,$prob\_cruza$,p)
    $p\_muta = 1 - p\_cruza$
    if random probabilidad $\leq p\_muta$ **then**
      function Operador Muta($Padres\_P\_0$)
        Seleccionar un genes random del cromosoma P_0
        return $Hijo \longleftarrow$ cambiar los genes
      end function
    else
      function Operador Cruza($Padres : P\_0$)
        Seleccionar aleatoriamente genes de los $Padres : P\_0$
        return $Hijo \longleftarrow$ combinación de los genes
      end function
    end if
  end function
  function Abortos(p)
    if suma(Hijo(binario)) == p **then**
      El hijo es factible
    else
      De nuevo GENERACION HIJOS($Padres\_P\_0$)
    end if
  end function
  function Remplazo(P_0,$prob_cruza$,p)
    Evaluamos el Fitness del la nueva población
    Remplazamiento del peor Fitness por el Hijo Factible
  end function
  function Busqueda Local($P's$,$tasa$)
    Evaluar los Fitness de la población de soluciones $Fitness(P's)$ y ordenarlas
    Dividimos la población según la tasa en $P's_{elite}$ $P's_{poblacion}$
    $\forall P's_{elite}$ intercambiar cromosoma de instalación $max(f_i)$ por inst. con $min(f_i)$
    if $Fitness(P's_{elite}) < Fitness(P's)$ **then**
      Intercambiamos en la población $P's \longleftarrow P's_{elite}$
    else
      Conservamos el $P's$ original
    end if
    Unimos los $P's$ que entraron a la búsqueda con el resto $P's_{elite} + P's_{poblacion}$
  end function
  Criterio de paro
  if *Min de los Fitness No cambia en la toleracia* **then**
    Detenemos el algoritmo
  end if
end for
return Mínimo del Fitness (F.O.), Solución Factible(y_i^*), Solución Asignación de los clientes

3. Generación de la Población

Para generar la población inicial, los cromosomas se generaron aleatoriamente. En el algoritmo propuesto se crea un vector de cardinalidad p donde cada entrada indica qué instalación es la que abre, ver el Cuadro 3.1. pero este es sólo para crear la población de soluciones iniciales pero sobre este vector no se pueden aplicar los operadores genéticos.

	Vector inicial		
Instalación.	2	4	1

Cuadro 3.1: En el supuesto de que $p = 3$, el individuo indica que se abren 3 instalaciones: la instalación 1, 2 y 4.

A pesar de que el vector anterior indica que instalaciones están abiertas no hay que perder de vista que la representación del conjunto de las instalaciones I está representado como el vector binario y_i que nos indica la localización, ver el Cuadro 3.2.

	Cromosoma				
Genes	1	1	0	1	0

Cuadro 3.2: Con base en el vector inicial anterior, con un conjunto de $I = \{1, 2, 3, 4, 5\}$ la codificación de las instalaciones abiertas está representada por el vector binario.

Finalmente, respecto a la representación de la solución, lo correcto es que represente la localización de las instalaciones porque los clientes se deben asignar a la instalación más cercana (que sería la más barata). Así al saber qué instalaciones están abiertas (localización) en el vector indicador y_i se puede saber la asignación de los clientes en la matriz indicadora X_{ij} según la matriz de costos C_{ij}. Por lo tanto, la codificación de la solución debe de ser del proceso de localización (debe indicar qué instalaciones están abiertas) y esta es binaria que es y_i y será el cromosoma sobre los que se aplicaran los operadores de cruza y muta.

4. Selección de padres

La selección de padre se realizo mediante un torneo de padres en el que se elige una vecindad de K individuos de la población de soluciones y se eligen dos individuos con los mejores Fitness (mejor valor en la Función Objetivo) para ser los padres.

5. Generación de hijos

Respecto a los operadores genéticos deben aplicarse a las soluciones de la población inicial. Para ello se usaron probabilidades complementarias conde la probabilidad de cruza es p_{cruza} y la de muta es $p_{muta} = 1 - p_{cruza}$.

5.1. Operador cruza

Para la cruza se elijen genes al azar de ambos padres y estos se combinan para crear un nuevo cromosoma. Si los hijos no son solución factibles entonces son abortos y entran a reparación.

5.2. Operador Muta

Para la muta se elige un gen al azar y se modifica.

6. Método de Reemplazamiento

Una vez evaluado el Fitness de la población se selecciona la solución con el peor resultado y se reemplaza por el hijo.

7. Búsqueda Local

La búsqueda local se aplica sobre una selección elitista. Por ejemplo, seleccionamos el 50 % mejor de la población total. Ésta población élite entra la búsqueda local en el que se intercambian los genes del cromosoma. A diferencia del operador muta en que el cambio de los genes es aleatorio; en la búsqueda local se intercambia la instalación ya abierta de mayor costo f_i con la instalación disponible con menor costo f_i. Sin embargo, el individuo nuevo sólo remplaza al original si el Fitness es mejor para la minimizacion y sí el individuo NO se encuentra ya en el pool de soluciones. De este modo, lo que se busca es ir refinando la población de soluciones de manera que se obtengan mejores valores en la Función Objetivo.

8. Criterio de Paro

La búsqueda local tiene como objetivo ir agregando calidad al pool de soluciones; sin embargo, para evitar hacer generaciones de manera innecesaria se estableció una tolerancia que indica las generaciones máximas sin cambio en la solución mínima encontrada. Tras superar la tolerancia el algoritmo se detiene y se toma la última solución óptima.

9. Datos de la Implementación

Los datos para la implementación se proporcionan en un archivo txt. Por un lado, se necesita el archivo contiene los valores de la matriz de costos para los clientes C_{ij}. Por otro lado, se requiere el archivo con los costos de localización de las instalaciones f_i. Los valores I y J deben corresponder a la detención de la matriz proporcionada. Mientras que el parámetro de las instalaciones a abrir p puede determinarse mientras que no sea más grande que el conjunto I.