METAHEURÍSTICAS

PRÁCTICA 1 - PROBLEMAS, INSTANCIAS Y SOLUCIONES. BUSQUEDA ALEATORIA

Universidad de Córdoba

Grado en Ingeniería Informática, 3er curso

Escuela Politécnica Superior

Miguel Morales Carmona

i12mocam@uco.es

45886851-B

CONTENIDO

DESCRIPCIÓN DE LOS PROBLEMAS

MAXMIN DIVERSITY PROBLEM

Este problema recibirá un número de elementos "m" que serán seleccionados de una matriz de distancias nxn, buscando que la menor de las distancias sea máxima al finalizar el algoritmo. La siguiente fórmula describe esta función objetivo :

$$z_{\mathrm{MMDP}}(\mathbf{x}) = \min_{i \neq j} \left(d_{ij} : x_i = 1 \wedge x_j = 1\right),$$
$$\sum_{i=1}^{n} x_i = m,$$
$$x_i, x_j \in \{0, 1\},$$
$$1 \leq i \leq n, 1 \leq j \leq n,$$

Estos elementos representan distancias que podemos encontrar en una matriz de distancias. Al terminar el algoritmo en una cierta cantidad de iteraciones debe haberse llegado a una solución aproximada.

TRAVELLING SALESMAN PROBLEM

El problema del viajero trata de obtener el recorrido más corto en un conjunto de ciudades, sin repitición y en un viaje cerrado. Para la solución del mismo se realiza un algoritmo en el que calcula las permutaciones de los nodos, calculando en cada iteración la distancia del viaje. Con las distancias en cada iteración se buscará minimizar todas esas sumas.

CUTWIDTH PROBLEM

Dispondremos de un grafo no dirigido que deberemos ordenar linealmente de tal forma que los cortes en las aristas sea mínimo. Por lo que probaremos permutaciones y minimizaremos ese valor de cortes. Siempre contaremos estos cortes cuando los vértices estén de izquierda a derecha.

CAPACITED P-HUB

Este problema consiste en una cantidad "n" de centros que pueden ser clientes o hubs. Tendremos un máximo de hubs y todos los centros son clientes a su vez. Cada cliente estará asociado a un solo hub.

Dada esta situación, el objetivo es minimizar la suma de las distancias entre clientes y sus respectivos hubs. Cada hub tiene unos recursos limitados, y es una restricción a tener en cuenta.

MAXMIN DIVERSITY PROBLEM

Este problema recibirá un número de elementos "m". Tendremos un vector binario que tendrá tantos unos como elementos "m". Se busca maximizar el mínimo entre la combinación de todas esas posiciones del vector solución. Por lo que con la siguiente función he realizado un vector booleano aleatorio en cada iteración :

```
función generarSolución
desde 0 hasta N (tamaño de la matriz)
    Solucion[i] <- 0
desde 0 hasta M (tamaño de la muestra)
    Si Solucion[Aleatorio()]==0 entonces
        Solucion[Aleatorio()]<-1
    Sino i--
```

Una vez tenemos el vector solución, realizamos todas las combinaciones posibles de esas posiciones buscando el mínimo de todas ellas.

```
función evaluarSolución
desde 0 hasta n
    desde 0 hasta n
        si Solución[i]==1 y Solución[j]==1 && i!=j entonces (Así controlo
        que no sea la diagonal principal y que sea una combinación posible
        del vector solución)
            si distancias[i][j] < mínimo entonces
                mínimo <- distancias[i][j]
devuelve mínimo
```

TRAVELLING SALESMAN PROBLEM

En este ejercicio contamos con una matriz de nxn, y generamos aleatoriamente permutaciones de esas posiciones para hallar una posible solución que debemos evaluar si es mejor que las de anteriores iteraciones.

función generarSolución

desde 0 hasta N

Solución[i]=Aleatorio()%N

Una vez con el vector solución evaluamos cuánto es el coste mínimo.

función evaluarSolución

desde 0 hasta N-1

mínimo=mínimo+distancia[solucion[i]][solucion[i+1]%(N-1)] (Así controlo el ciclo)

devuelve mínimo

CUTWIDTH PROBLEM

De manera parecida al ejercicio anterior, pero tendremos un vector de aristas o conexiones de tamaño nxn, debemos realizar permutaciones de n elementos para poder evaluar la solución.

función generarSolución

desde 0 hasta N

Solución[i]=Aleatorio()%N

función evaluarSolución

desde 0 hasta N

actual=Solución[i]

desde 0 hasta N

sig=Solución[j]

cortes=cortes+matriz[actual][sig]

si cortes > max entonces

max <- aristas

devuelve max

CAPACITED P-HUB

De nuevo nuestro vector solución será una permutación de conexiones entre clientes y hubs, mientras que para evaluarlo observaremos la matriz distancia para realizar el sumatorio. Al acabar cada iteración se evaluará si es una nueva mejor solución.

función generarSolución

Desde 0 a P (donde P son los concentradores o hubs)

VectorHubs[i]=PosiblesClientes[Aleatorio()%N-1)

Eliminar cliente de PosiblesClientes

Desde 0 a N

Asignar sin sobrepasar la capacidad de los hubs

Devuelve solucion

función evaluarSolución

Desde 0 hasta N

actual=Solución[i]

mínimo=mínimo+distancias[i][actual]

Devuelve mínimo

EXPERIMENTOS Y ANÁLISIS

MAXMIN DIVERSITY PROBLEM

Los datos han sido tomados de archivos de texto donde sus primeros dos valores eran n y m. Las líneas siguientes se forman de tres digitos; dos coordenadas y el valor de la distancia.

Valor máximo de la instancia más pequeña : 11.9381
Valor mínimo de la instancia más pequeña: 77.8342
Tiempo: .027255102
La instancia más pequeña es de tamaño 10. Su valor m es 3.

En cada iteración se acerca más a la solución mínima ya que prioriza la obtención del valor mínimo. La exactitud del algoritmo dependerá del número de iteraciones.

TRAVELLER SALESMAN PROBLEM

Los datos de la instancia han sido tomados de un archivo de texto donde la prímera línea era el valor de m y las siguientes los valores correspondientes a la matriz de conectividad.

Valor mínimo de la instancia más pequeña : 426
Valor máximo de la instancia más pequeña : 788
Tiempo : .039610092

El número aleatorio generado afectará en la bondad del algoritmo, así como a la dimensión del rango entre el máximo y el mínimo.

CUTWIDTH PROBLEM

Las instancias usadas contenían valores de n y m, estando duplicado el valor de n y en líneas siguientes del archivo de texto, encontrabamos las coordenadas y su elemento de conectividad.

Valor máximo : 6
Valor mínimo : 3
Tiempo: .029204607

CAPACITED P-HUB

Las instancias usadas para CPH consiste en un archivo que contiene en primer lugar los valores de n, p y c. Después de estos encontramos los identificadores de los nodos, coordenadas y sus demandas.

Valor mínimo de la instancia más pequeña : 1810.81
Valor máximo de la instancia más pequeña : 2847.47
Tiempo : .038336370
La instancia más pequeña es de 50 elementos.

En cada ejecución va cambiando la combinación de hubs y clientes y por ello el número máximo y mínimo oscila en un rango más o menos pequeño o constante que depende del tamaño de la instancia, del aleatorio y del número de iteraciones.