UNIVERSIDAD AUTONOMA DE NUEVO LEON.

FACULTAD DE INGENIERIA MECÁNICA Y ELECTRICA.

TAREA 6: ALGORITMO HEURISTICO PARA PROBLEMA DE TRAVELING SALESMAN PROBLEM

PRESENTADO POR:

JESUS JAVIER MORENO VAZQUEZ 1619830

HORA: V4-V6

PROFESOR: DR. MARIA ANGELICA SALAZAR

SAN NICOLAS DE LOS GARZA, NUEVO LEON, A 27 DE MARZO DEL 2016

**DESCRIPCION GENERAL DEL PROBLEMA**

Se tiene un conjunto de N nodos o ciudades, cada ciudad tiene una distancia o costo para llegar a otra ciudad, entonces, se requiere visitar cada una de las ciudades o nodos y regresar a la ciudad origen buscando siempre minimizar el costo o la distancia recorrida, dependiendo del caso.

Además, a la solución presentada se le tendrá que hacer un movimiento para intentar mejorar la solución.

**ALGORITMO PROPUESTO:**

1.- Lectura de Datos

2.- Calcular las distancias euclidianas.

3.- Calcular solución inicial con el método constructivo greedy o vecino más cercano, tomando como ciudad inicial siempre la primera.

\*\* Se había pensado sacar un promedio de las distancias de cada ciudad, y comenzar tomando la mejor distancia de la ciudad con el promedio más alto, sin embargo, los resultados no favorecían la utilización de este método. \*\*

4.- Calcular el tamaño de las aristas.

5.- Calcular la arista más pesada.

6.- Tomar dichos nodos (los de la arista más pesada)

7.- Realizar el movimiento: Se coloca la arista más pesada al inicio y se calcular el peso de la nueva posible solución, si es mejor, el contador de iteraciones se reinicia (0), si no es mejor, entonces se suma uno al contador y el segundo nodo de la arista se mueve un espacio hacia adelante hasta recorrer el tamaño de la lista de la solución, cuando llegue al final, se recorrerá un espacio el primer nodo mientras que el segundo se posiciona nuevamente en la primera posición de la lista solución. El proceso se detendrá cuando se realicen 1000 iteraciones sin encontrar un resultado mejor.

8.- Reportar resultados.

**RESULTADOS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Instancia | Distancia Calculada | Distancia Óptima | GAP | Tiempo |
| Eil51 | **513.610** | **426** | **20.56%** | **.095596** |
| Berlin52 | **8980.91** | **7542** | **19.07%** | **.092941** |
| Eil76 | **711.99** | **538** | **32.34%** | **.272756** |
| Rat99 | **1564.7248** | **1211** | **29.20%** | **.543158** |
| kroA100 | **26856.3885** | **21282** | **26.19%** | **.059207** |
| Eil101 | **825.2423** | **629** | **31.19%** | **.610044** |
| Lin105 | **20362.75** | **14379** | **41.61%** | **.701299** |
| Bier127 | **135751.7780** | **118282** | **14.76%** | **1.593606** |
| Ch130 | **7575.2862** | **6110** | **23.98%** | **1.274316** |
| kroB150 | **32825.7498** | **26130** | **25.62%** | **1.915692** |
| Ch150 | **8194.6143** | **6528** | **25.53%** | **2.007861** |
| kroB200 | **36981.59** | **29368** | **25.92%** | **4.512505** |
| Ts225 | **152493.5507** | **126643** | **20.41%** | **6.638751** |
| Pr226 | **94685.4541** | **80369** | **17.81%** | **6.076144** |
| Gil262 | **3241.46683** | **2378** | **36.31%** | **9.235933** |
| A280 | **3148.4173** | **2579** | **22.07%** | **12.238727** |
| Lin318 | **54033.5767** | **42029** | **28.56%** | **17.1445** |
| Pr439 | **131282.09** | **107217** | **22.44%** | **50.6998** |
| Dsj1000 | **24630960.1014** | **18659688** | **32.00%** | **643.69** |
| Pr1002 | **315596.58** | **259045** | **21.83%** | **709.5943** |

x = solución inicial

f(x) = Valor de función objetivo de x

N(x) = Vecindario

Repetir

x’ <- argmin { f(x´´)}

Si f(x´) <= f(x)

x🡨 x´

**CARACTERISTICAS DEL SISTEMA**

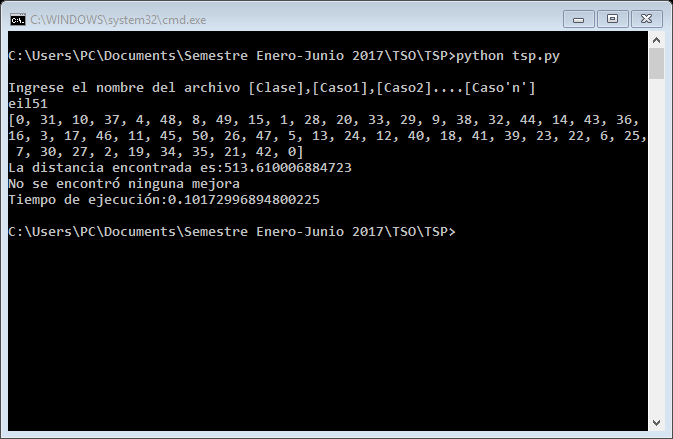
Procesador: Intel® Core i5-4590 CPU @ 3.30 GHz

Memoria Instalada (RAM): 8.00 GB (7.88 GB utilizable)

Sistema Operativo: Windows 10 Pro

Tipo de sistema: Sistema operativo de 64 bits, procesador x64

**Ejemplo**

****

**CONCLUSIONES**

Esta actividad me resultó más complicada que las anteriores, lo curioso es que tuve más problemas en codificar el método constructivo (vecino más cercano) ya que no cumplía con las restricciones, o tomaba un valor que no era el más cercano, dando resultados aún peores que los obtenidos. Al principio había intentado seleccionar mi ciudad de origen calculando los promedios de distancias de cada ciudad y tomando la más cerca de la ciudad con el promedio más alto (el mejor de lo peor) sin embargo, los resultados eran más altos a comparación de elegir siempre la primera ciudad, por lo que ese método fue descartado.

**BIBLIOGRAFIA**

* https://www.python.org/doc/