Curso: Heurística – CM0439
Profesor: Juan Carlos Rivera Agudelo
Julio 15 de 2024
Métodos constructivos y aleatorizados

METODOS CONSTRUCTIVOS Y ALEATORIZADOS

Teniendo en cuenta el problema asignado para trabajar durante el semestre (Vehicle Routing Problem with Time Windows – VRPTW –, ver Anexo 1), se deben implementar tres algoritmos: un algoritmo constructivo y dos algoritmos constructivos aleatorizados (construcción GRASP, construcciones con ruido, construcciones basadas en colonias de hormigas). Los algoritmos deben poderse parametrizar para ejecutarse con diferentes valores de cada parámetro según sea el caso (ver Anexo 4 – ejemplo de definición de parámetros).

El algoritmo debe ser desarrollado en Python o Matlab (otros lenguajes pueden ser utilizados sujetos a previa aprobación por parte del profesor).

Las pruebas del algoritmo deben ser realizadas con las instancias de prueba disponibles en las librerías para el problema seleccionado. Las instancias están disponibles en Interactiva Virtual con el formato descrito en el Anexo 2.

Cuando los algoritmos desarrollados sean basados en ideas de otros autores, i.e. método de los ahorros, NEH u otros, se debe indicar explícitamente en la descripción del método.

Formato de Entrega:

1. Realizar una presentación en diapositivas en la que se incluya:
    - Descripción de los algoritmos implementados.
    - Descripción de los resultados obtenidos, incluyendo:
        - Comparación con una cota inferior.
        - Comparaciones utilizando diferentes valores para cada parámetro.
        - Comparación entre los métodos implementados.
        - Tiempo de cómputo.
    - Conclusiones.
    - Se deben incluir todas las referencias bibliográficas que hayan servido de apoyo.
    - Se pueden incluir otras secciones en el informe.
2. Enviar por Interactiva Virtual los archivos correspondientes al código, los archivos de resultados y la presentación (con ayudas audiovisuales, formato libre).
3. Se realizará una sustentación oral individual en horario acordado con el profesor.

Se debe adjuntar los archivos correspondientes a los códigos de los algoritmos. Éstos deben poderse ejecutar sin necesidad de ingresar datos o modificaciones adicionales de forma manual. El algoritmo debe producir, para cada instancia disponible, un archivo de resultado con el nombre y formato descrito en el Anexo 3.

Curso: Heurística – CM0439
Profesor: Juan Carlos Rivera Agudelo
Julio 15 de 2024
Métodos constructivos y aleatorizados

Evaluación:

18%	Método constructivo
18%	Método aleatorizado 1
18%	Método aleatorizado 2
5%	Cota inferior
12%	Comparación de parámetros
12%	Comparación entre métodos
5%	Comparación de tiempos de cómputo
6%	Conclusiones
5%	Calidad de la solución
6%	Validación de resultados

FECHA DE ENTREGA: La fecha límite de entrega es el miércoles 4 de septiembre de 2024 (vía Buzón de Interactiva Virtual).

Curso: Heurística – CM0439
Profesor: Juan Carlos Rivera Agudelo
Julio 15 de 2024
Métodos constructivos y aleatorizados

UNIVERSIDAD EAFIT®

ANEXO 1

DEFINICIÓN Y FORMULACIÓN DEL PROBLEMA

In the vehicle routing problem with time windows (VRPTW), a large enough number of homogeneous vehicles must visit a set of geographically scatter nodes. Each vehicle can perform a route visiting a subset of nodes in such a way that each node must be visited exactly once by exactly one vehicle and each route must start and finish in a given depot node. The total number of required vehicles must be minimized, while solutions with minimal total traveled distance are preferred between solutions with the same number of required vehicles. Vehicles are characterized by a capacity, which limits their que total load, and while nodes must be visited within a time windows.

Problem statement

Given a fleet of K vehicles of the same capacity Q and cost M (M is a large value), the VRPTW is defined on a graph $G = (V; E)$, where $V = \{0; 1; \dots; n\}$ is the node set and $E = \{(i; j): i \neq j; i, j \in V\}$ is the arc set. The depot is denoted by 0 and $V_c = \{1; \dots; n\}$ is the set of customers. Each node i has a demand $q_i \leq Q$, a service time s_i and a time interval (window) $[e_i; l_i]$ in which service can start, $q_0 = s_0 = 0$ and $[e_0; l_0] = [0; l_0]$ is the time horizon for the depot. Time windows are considered to be hard, that is if a vehicle k arrives at node j earlier than e_j, the service is postponed until the opening of the time window, and arrivals after l_j are forbidden. A non-negative distance d_{ij} and travel time t_{ij} are associated with every arc $(i; j) \in E$. We assume that distances satisfy the triangle inequalities. A solution of VRPTW is composed of multiple routes, each one must respect the following constraints: 1) Each customer is only served once; 2) A vehicle k leaves each visited customer and travels toward the next one on the route and each route starts and ends at the depot; 3) The total demand on any route does not exceed Q; 4) Finally, the service at each customer must start within his time window and vehicles must return to the depot before l_0.

Curso: Heurística – CM0439
Profesor: Juan Carlos Rivera Agudelo
Julio 15 de 2024
Métodos constructivos y aleatorizados

ANEXO 2

FORMATO DE DATOS DE LAS INSTANCIAS

En el archivo "VRPTW Instances.zip" (disponible en EAFIT Interactiva) se encuentran 18 instancias con datos para el VRPTW, donde el número de nodos varía entre 25 y 100.

La primera fila indica el número de nodos de demanda (n) y la capacidad de los vehículos (Q).

Las siguientes $n+1$ filas indican, para cada nodo $i \in V$, el índice i, la coordenada x_i, la coordenada y_i, la demanda q_i, el límite inferior de la ventana de tiempo e_i, el límite superior de la ventana de tiempo l_i, y el tiempo de servicio s_i.

El primer nodo representa el depósito, origen y final de todas las rutas.

El tiempo de viaje entre nodos es calculado con base en la distancia euclidiana entre los nodos como

$$t_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Los valores de t_{ij} deben ser redondeados utilizando tres cifras decimales.

A continuación se presenta un ejemplo con $n = 6$ y $Q = 10$:

6	10					
0	10	6	0	0	40	5
1	6	2	3	0	15	5
2	3	9	5	10	20	5
3	3	6	2	5	15	5
4	1	0	4	15	25	5
5	7	8	1	0	10	5
6	0	0	4	10	25	5

ANEXO 3

FORMATO DE ARCHIVO DE RESULTADOS

Los resultados obtenidos se deben registrar en un archivo de Excel en el cual cada hoja contenga los resultados de cada instancia. El nombre del archivo debe ser VRPTW_<nombre_estudiante>_<método>.xlsx, donde <nombre_estudiante> debe ser reemplazado por el nombre (nombre, o apellido, o iniciales) de cada estudiante, y <método> debe ser reemplazado por un nombre indicador del método utilizado para hallar los resultados (constructivo, GRASP, ACO, LS, VNS, GA,…). Cada hoja del archivo de Excel debe ser nombrada con el nombre de la instancia respectiva (usar los mismos nombres utilizados en el archivo de datos, sin la extensión txt).

Los resultados deben tener el siguiente formato: La primera fila debe contener 3 valores: número de vehículos (m), distancia total recorrida y tiempo de cómputo (en milisegundos redondeado al entero más cercano). Las siguientes m filas (una para cada vehículo) debe contener el número de nodos a visitar (sin contar el depósito), la ruta recorrida por cada respectivo vehículo (lista ordenada de nodos incluyendo los depósitos), los tiempos de llegada a cada nodo, y la carga total del vehículo.

El siguiente es un ejemplo de una solución factible para la instancia de ejemplo presentada en el Anexo 2.

4	70.493	0							
2	0	5	3	0	0	3.606	13.078	25.078	3
2	0	1	4	0	0	5.657	16.042	26.859	7
1	0	2	0	0	7.616	20.232	5		
1	0	6	0	0	11.662	28.324	4		

En la solución anterior se usan 4 vehículos, con una distancia total recorrida de 70.493. El primer vehículo visita 2 nodos: parte del nodo 0, visita los nodos 5 y 3, y regreso al nodo 0. El tiempo de llegada a cada nodo es 0, 3.606, 13.078 y 25.078, respectivamente. La carga total del primer vehículo es 3 (1 + 2).

UNIVERSIDAD EAFIT®

ANEXO 4

EJEMPLO DE DEFINICIÓN DE PARÁMETROS

La siguiente figura muestra un algoritmo en lenguaje de programación Python donde luego de cargar librerías y funciones se definen los valores de los 4 parámetros a utilizar: nsol, alpha, K y r. En los algoritmos implementados los parámetros deben tener valores constantes; no deben ser leídos de algún archivo, ni solicitados al usuario en cada ejecución.

```
import xlwt
from xlwt import Workbook

from Constructive import Constructive

from GRASP1 import GRASP1
from GRASP2 import GRASP2
from Noise import Noise


# Data reading
from Read import read_fsspsc


wb = Workbook()
sheet1 = wb.add_sheet('Randomized')

nsol=100   # Number of solutions
alpha=0.05 # GRASP parameter
K=5        # Maximum RCL size
r=5        # Noise

for id in range(1,21):
    print(id)
    n,m,L,dur = read_fsspsc(id)

    Z0,S0,I0,F0,Fi0,t0=Constructive(n,m,dur,L,id)
    print("C:\t",Z0,"\t",t0)

    Z1,S1,I1,F1,Fi1,t1=GRASP1(n,m,dur,L,id,alpha,nsol)
    print("GRASP1:\t",Z1,"\t",t1)

    Z2,S2,I2,F2,Fi2,t2=GRASP2(n,m,dur,L,id,K,nsol)
    print("GRASP2:\t",Z2,"\t",t2)

    Z3,S3,I3,F3,Fi3,t3=Noise(n,m,dur,L,id,r,20)
    print("Noise:\t",Z3,"\t",t3)
```