Práctica 7. Problema de rutas de vehículos (VRP)

Belén Melián-Batista[a]

[a] *Universidad de La Laguna*
Departamento de Ingeniería Informática y de Sistemas
mbmelian@ull.edu.es
Curso 2021-2022

Objetivo:

Proponer, implementar y evaluar algoritmos constructivos y búsquedas por entornos para el Vehicle Routing Problem (VRP) con objetivo de minimización de la distancia total recorrida.

Tareas:

Además de las tareas descritas en el presente documento, los alumnos tendrán que realizar las modificaciones que se planteen durante la corrección de la práctica. Asimismo, tendrán que responder a un cuestionario de preguntas tipo test sobre los contenidos teóricos de los algoritmos constructivos y búsquedas por entornos.

entregas parciales y modificación final:

- Entrega parcial 1 (19 de abril): Voraz y fase constructiva de GRASP
- Entrega parcial 2 (26 de abril): GRASP con búsqueda locales
- Entrega final y defensa (3 de mayo): Práctica completa (incluye GVNS) e informe.

Evaluaciones parciales:

Código fuente y defensa del trabajo realizado: hasta 10 puntos; si el día de la corrección falta algún código o este es incorrecto, la práctica se calificará como No apta.

Evaluación final:

Código fuente y memoria: hasta 5 puntos; si el día de la corrección falta algún código o este es incorrecto, la práctica se calificará como No apta.

Modificación propuesta el día de la corrección: hasta 3 puntos.

Cuestionario sobre los contenidos teóricos: hasta 2 puntos.

Lenguaje de programación:

Java o C++.

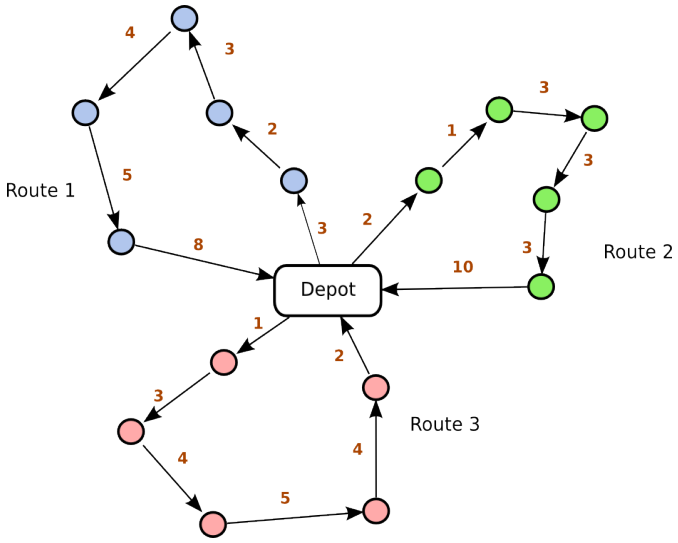
Ponderación de notas de las tres entregas:

Las defensas parciales supondrán un 25 % de la nota final de la práctica.

---

1. Vehicle Routing Problem con objetivo de latencia

En el problema de rutas de vehículos (VRP) hay un conjunto de clientes $\boldsymbol{V}$ que hay que visitar y un conjunto de vehículos $\boldsymbol{K}$ para realizar las tareas. Los clientes tienen que ser visitados todos, una sola vez. El número de vehículos disponibles será dado y se generarán tantas rutas como vehículos haya disponibles. El objetivo de este problema es visitar a todos los clientes minimizando la distancia total recorrida. En el ejemplo de la figura, la distancia total recorrida de la ruta 1 es igual a 25, la de la ruta 2 es 22 y la de la ruta 3 es 19. Por tanto, el valor objetivo de esta solución es la suma de estos tres valores $(25 + 22 + 19 = 66)$.



En el aula virtual disponen de las instancias de prueba que usaremos para el diseño e implementación de los algoritmos. El día de la modificación y defensa final, se proporcionarán otras instancias, de dimensiones similares.

Las tareas a realizar para el desarrollo de este proyecto son las siguientes:

1. Definir la codificación de las soluciones y la función de evaluación del objetivo (distancia total recorrida).
2. Diseñar e implementar un algoritmo constructivo voraz y un GRASP (para la primera entrega parcial, sólo se debe implementar la fase constructiva de GRASP).
3. Definir e implementar, al menos, cuatro estructuras de entorno diferentes (re-inserción, intercambio intra y entre rutas, 2-opt).
4. Diseñar e implementar un algoritmo de búsqueda por entornos variables general (GVNS).
5. Redactar el informe en latex (https://www.overleaf.com/) y compartir el enlace.
6. Realizar las defensas parciales en las fechas establecidas.
7. Realizar la modificación y defensa final el martes, 3 de mayo.

Qué debe presentar el alumno

a) Código fuente, debidamente comentado, y fichero ejecutable.

b) Una memoria en formato pdf en la que se describan brevemente los algoritmos diseñados enumerando las estructuras de datos usadas, las estructuras de entorno empleadas en las correspondientes búsquedas y cualquier elemento necesario para comprender el diseño propuesto.

c) Tablas o gráficas de resultados que muestren el comportamiento de los algoritmos sobre diferentes instancias del problema. A continuación, se muestra un modelo de tablas de resultados que el alumno puede usar. No obstante, se trata solo de un modelo que puede modificarse si se cree que otro es mejor.

Algoritmo voraz

Problema	n	Ejecución	$Distancia_Total_Recorrida$	CPU_Time
ID		1		
ID		2		
ID		3		
ID		4		
ID		5		
...	...	...	...	...

Cuadro 1: Algoritmo voraz. Tabla de resultados

GRASP

Problema	n	$\|LRC\|$	Ejecución	$Distancia_Total_Recorrida$	CPU_Time
ID		2	1		
ID		2	2		
ID		2	3		
ID		2	4		
ID		2	5		
ID		3	1		
ID		3	2		
ID		3	3		
ID		3	4		
ID		3	5		
...	...	...	...	...	...

Cuadro 2: GRASP. Tabla de resultados

GVNS

Problema	m	k_{max}	Ejecución	$Distancia_Total_Recorrida$	CPU_Time
ID		2	1		
ID		2	2		
ID		2	3		
ID		2	4		
ID		2	5		
ID		3	1		
ID		3	2		
ID		3	3		
ID		3	4		
ID		3	5		
...	...	...	...	...	...

Cuadro 3: GVNS. Tabla de resultados