## línea horizontal



Trabajo de Algoritmos: Algoritmo TSP GRASP

07/07/2022

**─**

Autores:

Nicolas Martinez, legajo: 45328.

Introducción

Dominio

El trabajo realizado se basa en el problema del Viajante de comercio (TSP por sus siglas en inglés). El algoritmo utilizado para obtener los posibles mejores caminos entre cada ciudad es el desarrollado a partir de la metaheurística de GRASP (Greedy Randomized Adaptive Search Procedure).

## Diseño

Inicialmente el proyecto se llevó a cabo generando una heurística Greedy para el problema de TSP. La heurística escogida es la del vecino más cercano (Nearest Neighbour). Esta heurística propone, de todos los vecinos actuales de la ciudad en la que se encuentra el viajante escoger el más cercano en cuanto al costo dado por las aristas del grafo. Una vez escogida la heurística se modificó para que los vecinos se recojan de manera aleatoria entre los mejores 5.

Posteriormente, se implementó la búsqueda local utilizando la técnica two-opt para poder obtener un mejor resultado de una colección de resultados obtenida a partir de sacar dos aristas e insertar dos aristas nuevas.

Finalmente, se implementó el algoritmo que se encarga de, por cada iteración procedural, buscar una solución, explorar sus vecinos con búsqueda local y guardar dicha solución en caso de ser mejor a la obtenida en la iteración anterior.

# Evaluación

Informe del Equipo:

* Microprocesador: Intel i3-9100
* Memoria RAM: 8gb DDR4 2400mhz
* SO: Windows 11

### **Gráfico:**

|  |  |
| --- | --- |
| Instancias | Iteraciones |
| burma14 | 5 |
| gr17 | 10 |
| bayg29 | 50 |
| brazil58 | 60 |
| eil76 | 60 |
| gr96 | 100 |
| eil101 | 120 |
| bier127 | 100 |
| ch130 | 160 |
| ch150 | 100 |
| brg180 | 150 |
| d198 | 150 |

# Análisis

Después de analizar el desempeño del algoritmo en varias instancias se comprobó que el algoritmo se comporta muy bien para instancias chicas sin embargo para instancias grandes se puede notar cómo varía mucho el comportamiento.

En promedio se necesitan al menos 100 iteraciones de GRASP para encontrar valores bastante buenos en instancias que superen los 100 nodos. Al estar randomizado el algoritmo a veces encuentra resultados cercanos al óptimo con pocas iteraciones y otras veces necesita al menos 100 iteraciones para poder encontrar algo bueno en términos de costos.

Algo para destacar que se vio en cada análisis es que la búsqueda local mejora enormemente el primer resultado del algoritmo Greedy. En cada iteración se puede notar como el valor del mejor resultado posible varía de manera grande, aunque después casi que se mantiene hasta pasar las 100 iteraciones en instancias grandes. Esto a veces causa que el algoritmo desperdicie tiempo de cómputo cuando la primera búsqueda local nos da un resultado bastante cercano al óptimo (o lo suficientemente bajo), entonces las demás iteraciones no encuentran nada mejor hasta la número 200 o 300.

Se estima que con 200 iteraciones de GRASP debería alcanzar para encontrar costos bajos, cercanos al optimo sin perder demasiado tiempo de cómputo (lo cual es inevitable).

# Conclusión

Como conclusión se puede decir que el trabajo fue mucho más entretenido de lo que pareció en un principio. El hecho de tener que probar y experimentar con los posibles resultados del algoritmo hace que se vea en acción todo el potencial de la meta-heurística cuando el primer resultado es lo bastante bueno. El modelado del algoritmo fue bastante tedioso al principio pero a la hora de escribirlo en código y ver cómo se comporta todo se hace mucho más sencillo (hasta cierto punto).

# Bibliografía consultada

* Rodríguez Peña, María Arantxa. (9 de Marzo de 2016). Problema del viajante de comercio con periocidad. Recuperado de: <https://riull.ull.es/xmlui/bitstream/handle/915/1982/Problema+del+viajante+de+comercio+con+periodicidad.pdf;jsessionid=3317C1BF1704B4E166FF980F2229F5B3?sequence=1>
* (11 de Agosto de 2017). Optimization with 2-OPT - Part 2 - Animated TSP Solver. Recuperado de: <http://pedrohfsd.com/2017/08/11/2opt-part2.html>
* Heurísticas Voraces. Recuperado de: <https://thales.cica.es/rd/Recursos/rd99/ed99-0033-04/voraz_viajante.html>
* Shrestha Bir Himalaya. (Junio 2021). Solving the Travelling Salesman Problem for Germany using NetworkX in Python. Recuperado de: <https://towardsdatascience.com/solving-the-travelling-salesman-problem-for-germany-using-networkx-in-python-2b181efd7b07>
* Interian Ruben. (Julio 2016).A GRASP heuristic using path-relinking and restarts for the Steiner traveling salesman problem. Recuperado de: http://www2.ic.uff.br/~rinterian/publications/journal\_papers/2017\_Interian\_SteinerTSP\_ITOR.pdf