Algoritmos Exactos y Metaheurísticas

Tarea 2

Víctor Reyes Rodríguez

April 19, 2023

Colonias espaciales 25XX

Es el año 2535, la humanidad ha aprendido a crear artificialmente atmósferas, por lo que las colonias espaciales están por doquier. Cada una de estas colonias requieren recolección de distintos tipos de minerales para así ir construyendo más infraestructura para sobrevivir. Para lograr dicho propósito, usted, jefe general de la autonomía de las colonias IO, Europa, Deimos y Titán, utiliza UAVs (*Unmanned Autonomous Vehicles*) de distintos tamaños, los cuales permiten la recolección de recursos. Dicho poder, conlleva la gran responsabilidad de organizar los aterrizajes y descargas de minerales, es decir, asignar a cada UAV un tiempo de aterrizaje determinado, para esto considere lo siguiente:

- Existirán un conjunto D de UAVs. Cada UAV tendrá un tiempo de aparición A_k, tiempo más temprano de aterrizaje E_k, más tarde de aterrizaje L_k y uno preferente P_k ($E_k \leq P_k \leq L_k$) $\forall k = 1, ..., D$. Existirá un tiempo de separación mínimo τ_{ij} entre el UAV i y un UAV j con $i \neq j$ (por la diferencia de tamaño entre ellos). Lo anterior implica, por ejemplo, que si el tiempo del primer UAV es T_i y del segundo T_j, con $T_i < T_j$, entonces $T_j \geq T_i + \tau_{ij}$.
- Por cada unidad sobre (bajo) el tiempo preferente P_k se penalizará en 1 el costo. El objetivo del problema es minimizar dicho costo.
- Considere que cada archivo es un caso de prueba distinto.

Se adjuntan 3 archivos (t2_Titan.txt, t2_Europa.txt y t2_Deimos.txt), los cuales siguen el siguiente formato:

```
Numero de UAVs
Para cada UAV i (i=,1...,D):
    T. mas temprano de aterrizaje , T. preferente ,
    T. mas tarde de aterrizaje (todo en la misma linea)
    Para cada UAV j (j=1,...,D):
        Tiempo de separacion requerido despues que i
        aterrice para que j aterrice
```

En base a lo anterior, se pide:

1. Diseñar e implementar (C/C++, Java, Python) un Greedy determinista y uno estocástico. En el caso del Greedy estocástico, muestre 5 ejecuciones (controladas por una variable seed) de su algoritmo para cada caso de prueba. Explique sus resultados. (**15 puntos**).

2. Diseñar e implementar un Hill-Climbing alguna-mejora y mejor-mejora sin restart. Utilice como punto de partida las soluciones generadas por ambos Greedy de la pregunta anterior. Reporte y explique los resultados. (**30 puntos**).

3. Diseñar e implementar un Tabu Search. Al igual que la pregunta anterior, utilice como punto de partida las soluciones generadas por ambos Greedy. Reporte y explique los resultados. (**15 puntos**).

Condiciones de entrega

- La tarea se puede desarrollar de manera individual o en grupos de a dos.
- La tarea se entregará vía Canvas de la sección. La fecha y hora límite para la entrega es el día Miércoles 10 de Mayo a las 23:59. No se aceptan tareas fuera de plazo.
- Además del código, debe incluir un informe en formato pdf, en donde deberá incluir pseudocódigos, experimentos y análisis.
- Consultas al correo `victor.reyes@udp.cl`, en mi oficina o en la clase.