0.1. Complexity analysis

These are the steps to solve the ACLP+RPDP with the complexity analysis:

0.1.1. Input data

Load K lines of text to compose a distances matrix for K airports, the m packed contents from previous nodes, and n items parameters from text files are of complexities $\mathcal{O}(K)$, $\mathcal{O}(m)$, and $\mathcal{O}(n)$, respectively.

Although there is some time complexity to load these input data into memory, we will consider them already loaded in memory, not including this parcel in the overall ACLP+RPDP solution time complexity.

0.1.2. Number of tested tours (T)

- If $K <= 6$, determine the number of tours through permutation. To get permutations of nodes as tours has a complexity of $\mathcal{O}(T = K!)$, as the nodes are permutated (factorial) to generate tours.
  - $\mathcal{O}(T = K!)$ or
  - $\mathcal{O}(T = 2)$, if the we are solving the two optimal TSP tours.
- If $K > 6$, we obtain 100 tours from a heuristic TSP solution (see its time complexity analysis in subsubsection 0.1.6):
  - $\mathcal{O}(T = 100)$

0.1.3. Solving T tours:

- $\mathcal{O}(T)$

a) For each tour with K nodes:

- $\mathcal{O}(T \cdot K)$

b) in each node we have to reset pallets destinations, to eliminate the risk of previous destinations assigned to a pallet, $\mathcal{O}(K \cdot m)$ and set the empty pallets destinations: $\mathcal{O}(K \cdot m + K \cdot m \cdot n)$

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n))$

c) To calculate the time limit to solve each node, it is necessary to determine the potential total volume for each node.: $\mathcal{O}(K \cdot n)$

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n + K \cdot n))$

d) The time complexity to put the packet contents destined to the next nodes in the tour on board: $\mathcal{O}(m)$.

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n + K \cdot n + m))$

d) The time complexity to optimize the pallets positions to minimize the CG deviation: $\mathcal{O}(m^2)$ (m pallets in m positions)

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n + K \cdot n + m + m^2))$

e) Plus the time complexity of the method used to solve an ACLPP with the aircraft partially loaded: $\mathcal{O}(C)$

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n + K \cdot n + m + m^2 + C))$

0.1.4. Considering the Shims algorithm: $\mathcal{O}(C) = \mathcal{O}(Shims)$

a) the time complexity for sorting the pallets in non-descending order of distances to the CG:

- $\mathcal{O}(Shims) = m \cdot log(m)$

b) As each pallet is solved at a time, the overall Shims complexity is multiplied by m.

- $\mathcal{O}(Shims) = m.(m \cdot log(m))$

c) The greedy phase has $\mathcal{O}(n)$.

- $\mathcal{O}(Shims) = m \cdot (m \cdot log(m) + n)$

d) The composition phase has $\mathcal{O}(n)$ added to the First-Fit Decreasing algorithm time complexity $\mathcal{O}(m \cdot log(m))$

- $\mathcal{O}(Shims) = m \cdot (m \cdot log(m) + n + n + m \cdot log(m))$
- $\mathcal{O}(Shims) = m \cdot (2n + 2 \cdot m \cdot log(m)) = m \cdot (n + m \cdot log(m))$

The overall time complexity to solve the ACLP+RPDP with Shims is:

- $\mathcal{O}(T \cdot K \cdot (K \cdot m + K \cdot m \cdot n + K \cdot n + m + m^2 + m \cdot (n + m \cdot log(m))))$

0.1.5. Complexity analysis considerations

This complexity analysis indicates an algorithm with nested dependencies on several variables (T, K, m, and n):

- $T \cdot K$: repeated iterations T for each of K nodes, nested loops where inner operations are contingent upon K.
- $K \cdot m + K \cdot m \cdot n + K \cdot n$: These operations scale with both K and each of m and n, indicating processes that involve all pairs of K with m and n.
- $m + m^2$: operations with linear and quadratic dependencies on the number of pallets m, indicating that as m grows, the complexity increases more significantly due to the quadratic term m^2.
- $m \cdot (n + m \cdot \log(m))$: Combines linear scaling with m and n and a logarithmic term $m \cdot \log(m)$, because the algorithm uses divide-and-conquer strategies on m elements, the Shims heuristics.

The quadratic (m^2) and logarithmic ($m \cdot \log(m)$) terms indicate that the algorithm's efficiency decreases significantly as the number of pallets (m) increases.

The overall complexity suggests an algorithm that performs efficiently with small values of K, m, and n but may become substantially more resource-intensive as these parameters increase. The number of tours T indicates that the entire complex operation is executed T times, further amplifying the effect of the inner complexities.

Optimising such an algorithm involves minimising the repetition count (T), reducing the complexity of operations dependent on K, m, and n. As a suggestion to deal with this (reduce the number of tours to be tested), we propose a TSP solution with the Genetics Algorithm, whose complexity is analysed in sub-subsection 0.1.6.

0.1.6. Time Complexity Analysis of the Genetic Algorithm for TSP

When $K > 6$, we obtain 100 tours from a heuristic TSP solution with the Genetic Algorithm (GA), which addresses the TSP by evolving a population of candidate tours through selection, crossover (recombination), and mutation over multiple generations, aiming to minimise tour length.

The number of nodes (K), population size (P), number of generations (G), and complexity of the crossover (C(K)), mutation (M(K)), and selection (S(K)) operations all have an impact on the method's time complexity.

Given these parameters, we can express the overall time complexity as:

- $\mathcal{O}(G \cdot (P \cdot (S(K) + P \cdot (C(K) + M(K)))))$

The selection complexity, $S(K)$, depends on evaluating the path length of each tour. The complexities of the crossover and mutation operations, $C(K)$ and $M(K)$, are functions of K because they involve manipulating sequences of nodes.

Each generation involves selecting tours based on shorter distances, generating new tours through crossover, and applying mutations, leading to the multiplication by G.

The direct involvement of K in the complexities of $C(K)$, $M(K)$, and $S(K)$ highlights how the number of nodes affects the algorithm's performance. Specifically, operations that manipulate or evaluate paths become more complex as K increases.

The encoding of tours and the calculation of path length have greater influence on the GA's computational demands.