Describe clearly with natural language step by step your(s) chosen solution(s).

In the proposed problem, we have an input being a list of three-dimensional allocatable spaces (containers) that are supposed to be merged in order to optimize the allocation options. In other words, we must merge the containers in order to produce larger containers. That is translated into an objective function that aims to maximize the average container volume in the list.

This merge operation is subject to some constraints. The result of this process must produce containers that cover the same points in space. The merge can only be performed when the basis of the container is in the same height. There should not be any container in the output that is totally contained in other container. Finally, the number of containers in the output should not be greater than three times the number of containers in the input.

The approach I used to tackle this problem was to first understand what this merge operation could be and try to figure out when the merged containers are better than the parent containers, when should we keep both and when the merged container is the best and we should discard the parent containers. To break down the problem into a simpler one, I have performed this analysis on pairs of containers. So, I have analyzed which merged containers could be proposed from a pair of parent containers depending on how the parents containers are in relation to each other. The proposed merged containers were only considered when their volume was greater than the average of the parent containers volume, since that would guarantee that we are increasing the overall average volume of the containers.

To solve the big problem from the small one. I have chosen to iterate over the containers in a recursive way performing these merge operations (in pairs) when possible. This recursive approach would end when the number of containers reach three times the initial one, or not any merge was done in a complete iteration. And then, the output containers would be the result.