OUR NEW PROBLEM:

We have already studied the SSM-KR problem in the linear route. Now, on this basis, we are studying the SSM-KR problem on the annular route. However, the meaning of K on the annular route changes and it represents the number of given rehandles for every layout.

Similar to the study of linear route, we also designed an approximate algorithm to solve the loading plan in the annular route. Nevertheless, the instances are different from the previous research. For example, we have expanded the number of containers to twice times the size of the original.

The key to turn the problem in the linear route into a circular route is to grasp the connection process of the last port in each loop and the starting port in the next loop.

Unlike straight lines, there are many other questions to think about in the annular route, such as whether the layout of the ship has some regularity after the end of each loop, and the variation of the number of used stacks in each port in each loop. That is to say, we should take the circumstances that the same ports in every loop and the different ports in the same loop into consideration.

CURRENT WORK:

Generated lots of new instances. The previous instances are not able to represent containers that loading at current loop, but unloading at the next loop. The new instances can indicate that the container is shipped in the current loop, and discharged in the next loop. For example, container (4, 2) means it should be loaded at port 4 in the current loop and unloaded at port 2 in the next loop.

The different representations of containers in the adjacent loops are considered. In the linear route, there is no need to consider the procedure that from one loop to the next loop. In a circular route, a containership calls at ports repeatedly. We cannot change the information of each container since we need to use the information in each loop. In order to facilitate the handover process of containers in different ports, we use different variables to temporarily replace the information in the container.

In order to facilitate the analysis of the problem, we divided into three kinds of scale instances according to the combination of parameters: large, medium and small. We can analyse the similarities and differences between different instances in their results.

CURRENT RESULTS

For all instances with the value of K equals 0, the number of used stacks in the last two ports in every loop is same and the number of used stacks in the rest ports converge to the same value after the second loop.

For small instances that H is selected from {4} and N is selected from {100}, the result is the same even though the instances have different values of limit height, seed and K.

For medium instances that H is selected from {8} and N is selected from {400,1000}, the result of most instances is the same even though the instances have different values of limit height, seed and K. There are some exceptional instances, the number of used stacks has a slight change after calling at several ports.

For small instances that H is selected from {12} and N is selected from {2000}, the result is the same even though the instances have different values of limit height, seed and K.

In short, the number of used stacks in the same ports in each loop can converge to the same value and the layout of a container ship in the same port in each loop should include containers with the same information.

INTRODUCTION

Recent increases in the demand for maritime transportation of ontainers have led to the construction of numerous mega container vessels, many of which have the capacity to carry more than 18,000 twenty-foot equivalent units (TEU) containers. A stack is obtained by stacking containers vertically. Several stacks in a row form a bay, and bays are placed side by side to form a container block.

In a general case, container ship serves many different ports on each voyage. A stowage planning for container ship made at one port must take account of the influence on subsequent ports. So the complexity of stowage planning problem increases due to its multi-ports nature. This problem is NP-hard problem.

PROBLEM DESCRIPTION

In the CSSM, the whole route of the ship is investigated; in particular, at each port of the route different sets of containers must be loaded for being shipped to the next ports. at each port the sequence of two handling operations affects the effectiveness of a stowing plan: first, the import containers must be unloaded from the ship, then the export containers can be loaded.

Given a ship with its structural characteristics, its route, described by a circular sequence of ports to be visited, and its current cargo, the problem consists in defining the stowage plan for a given set of containers that differ for the loading and destination port, so that all the containers are loaded on board, while the structural and operative constraints are satisfied and the number of stacks used on the vessel at the ports for loading/unloading operations is minimized.

Each ship travels on a circular route with *P* ports and the transport demand is randomly generated in such a way that for each origin o in the route of a ship. For example, if *P* is equal to 6, when planning the stowage for port 5, the ship has on board a cargo deriving from loading operations executed at ports 2, 3 and 4. At port 5, after that the unloading operations are executed, the loading process regards containers bound for port 6, 1’, 2’, where 1’ and 2’ denote the port 1 and 2 reached during the second round of the ship. For each containership, five instances have been generated; each instance differs from the others for the transport demand to satisfy. Anyway, all instances have been generated in such a way to stress the capability of the heuristic approach to obtain feasible and effectiveness solutions in a short amount of time. We divide our instances into three parts according to the size of limit height: Small Ship with H equals to 4 and N is selects from {100}; Medium Ship with H equals to 8 and N is selected from {400, 1000}; Large Ship with H equals to 12 and N is selects from {2000}.

There are some assumptions in the following:

1. All the containers have the same size, 20 feet container;
2. Containers at each port for every loop are the same numbers and types;
3. The limit height of the vessel is fixed;
4. After the unloading and loading operation at each port, there are at most K re-handles exist;
5. The voyage should stop at a certain loop.
6. Other constraints to make sure security are satisfied.

OUR ALGORITHM

Our constructive heuristic algorithm is easy to understand since it is similar to the greedy algorithm. There are main three procedures: unloading, sorting and loading. Different from the SSM, there are different containers on the vessel at each port for C-SSP. Due to the characteristic of circular routine, we have to design more data structure to store different container types and Simulate different loading and unloading situations.

Considering the existing of re-handles, our strategies to handle loading and unloading operations are different. The rules become strict when it comes to meeting the conditions that there are certain re-handles given. However, the main ideas are extremely similar. The stacks on the vessel are divided into different categories according to whether it will occur re-handle when the loading container is loaded into the stack. Hereafter, we give them different priority to reduce the number of used stacks as much as possible. The priority of them is following:

1. Partial stacks with no re-handles occurring.
2. Partial stacks with re-handles occurring.
3. Empty stacks.

Obviously, there won’t be the second one if the given re-handles run out.

ANALYSIS IDEAS

1. First, get the average result with different sizes of instances respectively.

We get the conclusion that given certain re-handles can reduce the number of needed stack.

1. The number of used stacks and the size of instances are proportional when we combine with three different sizes of instances.
2. To get more interesting founding, we have a deep analysis of experimental data. We observe how the number of used stack at a fixed port change as the vessel visits all the ports in the circular routine. In addition, we can investigate the tendency of a specific port with the all number of used stacks in different loops.
3. Focus on some exceptional instances. Exceptional instances represent the particular situation that hardly can happen. Through the deep analysis of exceptional instances, we can make our research more fruitful and credible.
4. Make some charts by using results to help readers easily understand our article and the visual interface add a lot of highlights to your article.

CURRENT FINDINGS

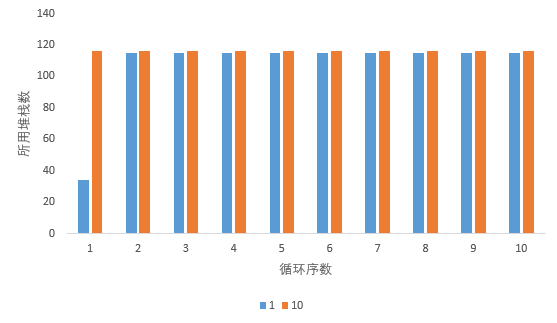
1. The last port and the second last port in the first loop converge faster than other ports. The theoretical guarantee should be proved.
2. Apart from minority of instances, the number of used stacks at every port in most instances converge to a certain value and stop changing.
3. The distribution of the number of used stacks of multi ports in the first loop is extremely different from the second loop. After a certain number of loops, the distribution remains unchanged.
4. For some instances, the number of used stacks in every port can converge faster if there is no re-handle. When giving a certain number of re-handles, layouts will remain unchanged after volatility.
5. Instances with the same parameters but different seeds have a strong influence on the number of used stacks. What’s more, the influence is

dynamic with the different parameters.

Proposition 1: The last port and the second last port in the first loop converge faster than other ports. Theoretical guarantee is following:

The last port and the second last port, no matter what the loop is, the containers at port are the same because there are not containers from the last loop in their layouts after unloading and loading operations all the time. However, the other ports are different since the layout in the first loop is different from the layouts in other loops. Hence, the last port and the second last port in the first loop converge faster than other ports is proved.

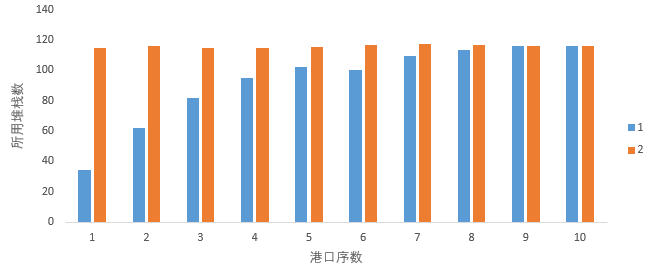
As is shown in the following figure, the blue stands for port 1 and the orange stands for port 10 (P=10). The number of used stacks at port 1 converges in the first loop and at port 10 in the second loop.



Proposition 2: The distribution of the number of used stacks of multi ports in the first loop is extremely different from the second loop.

It is not difficult to draw this conclusion. In the first round, there is not containers from the last round at all ports. After the first round, the layouts of some ports contains two types of containers and the number of used stacks increases.

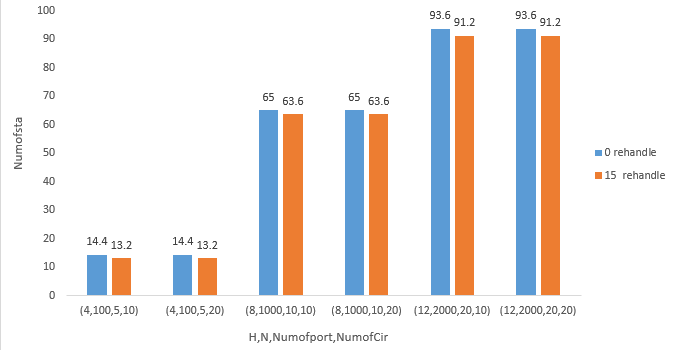
As is shown in the following figure, the blue stands for the first loop and the orange stands for the second loop.



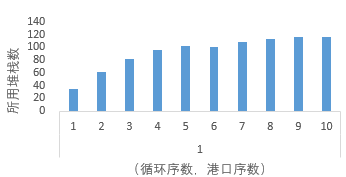
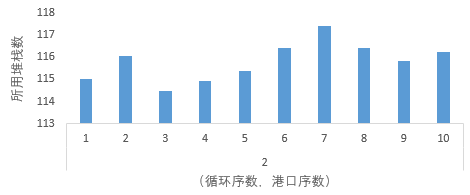
Proposition 3: the number of used stacks is no greater than .

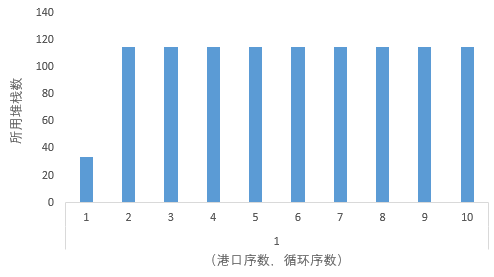
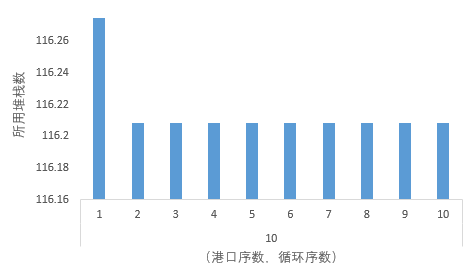
Considering the characteristics in the circular route, the upper bound of our problem becomes since the number of containers at a single port can’t reach 2\*N.

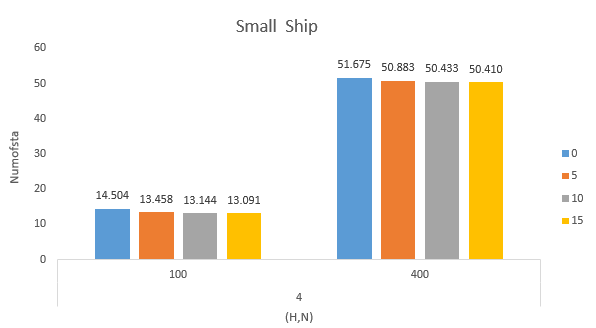
The following figure shows us the basic conclusion from our experiments. Given a certain number of rehandles do reduce the number of used stacks, especially for each layout in the circular route. In additions, it is proved that dividing the instances into three kinds is reasonable.

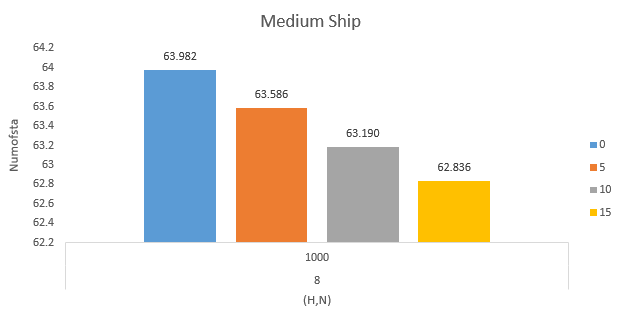


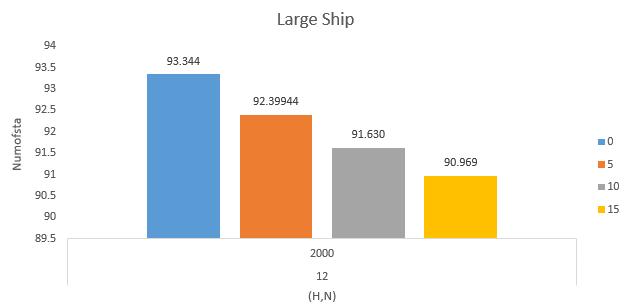
Another picture can be useful.







LR about circular routes:

Ambrosino et al. (2015) proposed a Mixed Integer Programming (MIP) heuristic aimed at determining stowage plans in circular routes for container ships so as to give support for the ship coordinator and the terminal planner.

To the best knowledge of us, it was the first time circular routes had been considered into this kind of problem.

Numofport

Figure. Results of different sizes and rehandles

Figure. Convergence of port 1 and port 2

Figure. Convergence of port 9 and port 10

Figure. Results under different loops

Figure. Results of different instances

Figure.