Dear Editor and Reviewers,

We would like to express our gratitude for the great efforts that you and the anonymous referees have put in this paper. We found most of the critiques and feedbacks very constructive. We have strived to address all the issues as thoroughly as possible. Consequently, we have rewritten many sections of the manuscript. The main changes are summarized as follows:

1. We change the title of our paper to ”A feasibility-based heuristic for the container pre-marshalling problem”. Words “generic” and ”scheme” are removed. As presented in point 4, the original idea is to separate framework and techniques, so that future scholars can take any part as they want. But we find it’s hard to follow, hence Section 5-7 are compressed into one Section, which explains the heuristic as well as associate techniques. “Generic” and “scheme” now are useless.

2. We have consolidated Section 3 and 4 into Section 3. The new Section 3 defines the problem (in Section 3 of the old version) and gives problem-related notations (in Section 4 of the old version). Solution-related notations (in Section 4 of the old version) are moved to the algorithm and will be introduced when used.

3. We add a new Section 4 to introduce state, concepts and properties of states. “State” is an important concept for the algorithm development; in the meanwhile, the analysis of state properties is useful for further study on algorithms. Hence, we write the contents in an independent section.

4. We compress Section 5-7 into Section 5. The initial thought for us is to propose a framework and several techniques. To introduce them separately is to emphasize that every part can be used solely. Other scholars can use the framework combined with other techniques; the proposed techniques can also be used in other frameworks. Hence, Section 5 explains framework, Section 6 explains techniques and Section 7 illustrates the assembled heuristic. However, we find such a presentation is too hard to follow. Now we just explain the assembled heuristic, and the details of the framework and techniques are introduced after the heuristic in the same section (Section 5). Useful parts will still be adopted if some reader thinks that they are effective in the future research.

5. Most of the presentation in this paper is rewritten. Figures and examples are added for better understanding. For new ideas such as the tier-protection indicator and task types, we add some explanations to reveal the thinking process.

We hope our revision meets your satisfaction, and we look forward to your favorable response.

Regards,

Bo Jin

Andrew Lim

Ning Wang

Zizhen Zhang

Reviewer1

1. The authors need to clearly demonstrate adequate understanding of the relevant literature. The authors need to include a good literature survey to show exactly what is novel about their approach. In this regard, I would like a clear discussion on the current literature versus the contributions of the paper. Now, only a few references are considered in the literature review.

To the best of our knowledge, only seven works discuss the pre-marshalling problem. These seven works have been discussed in our revised version (Section 2). The contributions (innovations) have been listed in the last but one paragraph of Section 1: This paper makes two main contributions to the literature. One contribution is the idea behind the proposed heuristic. In the traditional algorithms, containers are allocated according to a certain order, such as the descending order of their groups [8]; in this research, the order to allocate containers is not fixed beforehand, but rather that it is decided as the algorithm goes on. This innovation undoubtedly enlarges the search space and decreases the search efficiency. To avoid efficiency decrease, we propose the concept of state feasibility to check the feasibility of allocating a certain container before we actually allocate it. The time complexity of check feasibility is only O(G), here G is the number of groups. Our paper is the first work that uses feasibility to cut branches in CPMP algorithms. Another contribution is that the proposed heuristic performs outstandingly on both dense and loose instances, compared with existing heuristics.

1. In the conclusion section, the authors need to clearly discuss their theoretical contributions in Operations Research compared to those in related papers found in the literature. Additionally, the authors need to discuss and supply at least 4-5 solid and insightful future research directions for the operations research community.

The conclusion has been revised, and we have discussed our contributions as well as the future research road.

1. I would like authors to add a paragraph in the introduction discussing the optimization problem from an "operations research" point of view. The introduction does not clearly place the paper in that area.

We have revised the introduction session. The formal definition of the problem talked about is given in the last paragraph of Page 2, which infers that the problem is a typical optimization problem.

1. After reading the paragraph dedicated to describe the contributions of the paper in the introduction, I think that most of the readers could not to obtain a clear idea of what the paper proposes. Specifically, only a set of novel terms are provided with no context in the research field. Consequently, the contributions must be clearly introduced.

The new contributions are illustrated in the last but one paragraph of Section 1 as well as the answer to Question 1.

1. It would be desirable to provide some including remarks to further discuss the proposed methods, for example, what are the main advantages and limitations in comparison with existing methods?

We have added our innovation in contribution part of Section 1 and comparison with other algorithms in Section 5.

Briefly speaking, our algorithm is a kind of target-driven algorithm. This kind of algorithm has been talked about in some extant works. Target-driven algorithm rearranges a certain container to its final slot by a series of movements, and the algorithm does this until all containers are rearranged. By far, target-driven is effective compared with other kinds of algorithm. On top of this, our algorithm optimizes the rearrangement order, and series of movements for rearranging one container. The basic idea is to use state feasibility to cut searching branch. Now our algorithm is the best among target-driven heuristics.

1. The abstract in current form is superficial. Specifically, the abstract must be rewritten to point out significance and impact of the paper for EJOR. For instance, check the following phrases: 'We propose a generic feasibility-based heuristic scheme for the CPMP, which is a departure from the traditional heuristic strategy. An implementation of the heuristic scheme, the greedy and speedy heuristic, is also proposed, in which four well designed heuristic techniques are applied.' I wonder why the scheme is generic? are other proposals that consider infeasibility? why is a departure from 'traditional heuristic strategy'?, etc. Answers for these and similar questions are not provided in the remainder of the paper.

We have reorganized the abstract. The key ideas of our algorithm are emphasized and explained.

The ‘generic’ selling point in the old version is to emphasize that our framework and techniques for container rearrangement can be separately used by other scholars. But now we do not emphasize their independence, so we remove the word “generic”. Other proposals have never used feasibility to cut branches before. “Departure from traditional heuristic strategy” means we use feasibility which is different from others. The presentation makes the reviewers confusing, we have revised.

1. Moreover, the paper does not clearly communicate neither what has been done nor what is its significance. It is extremely hard to follow the description of the heuristic scheme without providing illustrative examples and a suitable introduction of the notations used along the whole paper. In this regard, the authors must to carry out a great effort in communicating their proposal clearly. I would say that in general it lacks many explanations and contextualization of the scheme.

What has been done: we propose a heuristic better than other heuristic and the significance lies in the idea behind the heuristic and the details such as state feasibility and tier-protection. We have stated this in Section 1 and 5.

We have rewritten the heuristic. We first add Section 4 to explain necessary concepts such as state feasibility, stability, etc. Then we consolidate Section 5-7 in the old version into one section (Section 5). The new section emphasizes the target-driven heuristic, which is composed of task selection and speedy task accomplishment procedure. The details of the two components are also explained in Section 5.2-5.4. Necessary examples and figures are added for better understanding.

1. I do not like the order of sections 3 and 4. Introducing notations concerning a problem that is not yet presented is very confusing. The reader must perform an effort to devise concepts such as 'order-p demand', 'interim stack', and so forth.

We remove Section 4 (Notations). For problem-related notations, we introduce them when introducing the problem in Section 3. For notations used for problem solving, we explain them when they are used.

Reviewer2

1. p. 7, l. 3 The authors should explain in greater detail what unreachable tiers are.

We have explained unreachable tiers in Page 11. Unreachable tiers are tiers that any movement sequence cannot reach. For a stack *s*, only the top containers can be moved to other stacks. The immovable number of containers (unreachable tiers) for any stack is .

1. p. 8, l. 36, p. 9, l. 4-6 Before giving a proposition on necessary conditions for state feasibility the authors should define the term "feasible state". The notion "feasibility of stack(s)" must also be defined. The first sentence on p. 9 reads as an unproven claim.

We add the definition of feasible state in Definition 3. A state (*L*,*f*) is feasible if there exists a movement sequence that can convert *L* to an orderly layout without moving fixed containers indicated by *f*.

“feasibility of stack(s)” is a typo, and it should be “feasibility of state”. But the sentence has been removed from the paper.

The first sentence on p. 9 is the conclusion from “Wang, N. (2014). Optimization Study on Container Operations in Maritime Transportation. PhD thesis, City University of Hong Kong, Hong Kong” which gives the proof to this claim. The work can be retrieved from City University Libarary->E-resources->E-theses->By title, and we are preparing a paper for this claim. In this paper, we do not need sufficient conditions, we only need necessary conditions. Hence, we explain the necessary conditions on Page 7 and 8. The first sentence on p. 9 is removed.

1. p. 9, l. 42 Proposition 2 should be proved and illustrated by an example.

We have added the explanation and examples on Page 10. When E ≥ 2(H − 1), it has N ≤ SH − 2H + 2, even though all containers are fixed, there are at most S − 2 stacks fully occupied with fixed containers: 1) if there are fewer than S-2 stacks fully occupied, the state is not a dead-end state; 2) if there are S-2 stacks fully occupied, there are at most two extra containers in the left two stacks. The state is not a dead-end state, either. For cases with E<2(H-1), there exist possibilities that the algorithm falls into a dead-end state.

1. p. 10, l. 25 Proposition 3 should be proved and illustrated by an example.

Proposition 3 does not have relationship with the content in this paper, therefore, we remove the proposition.

1. p. 12, I. 21-22 Please improve the formulation "If a container stack pair leads to an extreme state ..." since in my opinion rather the S-2 fully fixed stacks lead to an extreme state.

The meaning of the quoted sentence is “if the resultant state after performing a container-stack pair is an extreme state…”. We rewrite this part on Page 13, and now the sentence is removed.

1. p. 12, l. 29-31 In the definition of a pre-tricky state the last condition (Sum of f(s) < N -2) should be motivated.

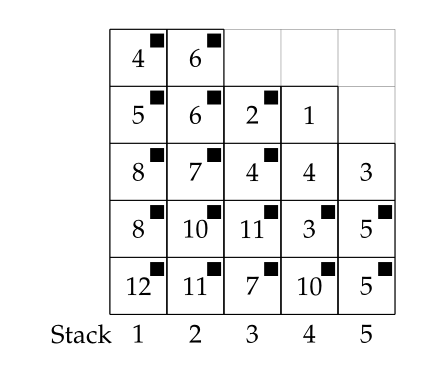
We have rewritten Definition 7. In the new definition, we discuss the resultant layouts after performing a task c→s with *f*(*s*)*=H-*1. The resultant layouts are different when ,,, and . It shows that only when , the resultant layout has a chance to generate dead-end states.

Moreover, we change the name of “trick state” to “dead-end state”.

1. p. 12, l. 40 ff This paragraph is hard to understand. Maybe an example would help.

We further explain the tier-protection indicator on Page 14 and 15. The tier-protection indicator is to balance the trade-off between the freedom of task selection and the surplus loss. In principle, a container can be moved to any valid stack. But if a container with small group value occupies an (relatively) empty stack, then the loss of the surplus vector is large. After the target container *c* is fixed to the target slot (*s*, *f*(*s*)+1), the surplus ∆(ϕ) is reduced by *H*− *f*(*s*), for *ϕ* ∈ (*g*(*c*),*g*(*s*,*f*(*s*)]. The larger the spread between *g*(*c*) and *g*(*s*,*f*(*s*)) is, the more the surplus vector will lose.

As shown in the following example, if container 1 (container with group value 1) is moved on container 3 (stack 5 tier 3), the supply of group 3 is reduced by 2 as 1 becomes the highest orderly container of stack 5. The cumulative supplies G(3) and G(2) are reduced by 2 accordingly.



1. Numerical experiments The authors should provide runtimes for all compared solution methods and for the CVS instances as well as the BF instances (even if the run times for GASH are very small). In the case of the BF instances they should include the heuristic by Bortfeldt and Forster (2012) in the comparison.

We add the runtimes of the three algorithms for both CVS and BF data sets. The algorithm of BF is a tree search; therefore we don’t compare it with our algorithm.

Minors:

1. p. 8: f ≤ 0 or better f = 0 ?

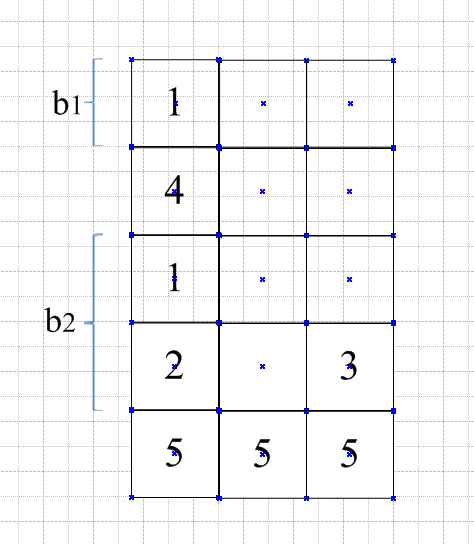
In the old version, ***f***≤***o***, here ***o*** is the vector composed by numbers of orderly containers in stacks. In the new version, we have defined that fixed containers are orderly and not allowed to be moved in the future. Hence, f≤o is useless and removed.

1. p 10: "solubility" - write "resolvability"

Proposition 3 has been removed since it has no relationship with the main contents of the paper. We have illustrated and explained it in “Wang, N. (2014). Optimization Study on Container Operations in Maritime Transportation. PhD thesis, City University of Hong Kong, Hong Kong” and our another paper.

1. p 15: "blocking containers above and below c\*" - blocking containers below?

We have explained blocking containers on Page 16. Blocking containers refer to those above target containers or target slots. In the internal task, c\* (s,t) is moved to a slot f(s)+1 under c\*. Hence, the number of containers above c\* is b1, and the number of containers between tier t (excluded) and f(s)+1(included) is b2. In the following example, container with group label 4 located at (1,4) needs fixed into stack 1. b1 and b2 containers are marked by brackets.



1. p 16: "realized implementation" - "implementation" would suffice.

The new version has removed the statement.

Reviewer3

1. Improve and ease the presentation, especially Section 6. Maybe, including examples and pictures might help in achieving this goal. Similarly, while the precision and rigor of Algorithms 1-6 is appreciated, the author should rethink the way in which the ideas behind some of them (especially Algorithms 4 and 5) is currently presented.

We have moved the contents in Section 6 to Section 5.4. We add examples and figures for bettering understanding, such as the examples for pre-extreme states and tier-protection.

The main content is to accomplish a task based on different types. The algorithms for internal and external types are presented in Algorithm 2 and 7, respectively. In Algorithm 3-6, they present all the functions appeared in Algorithm 2 and 7. As the functions are many, we explain them one by one, especially the motivations and advantages.

1. Proposition 2 at page 9 on the existence of "tricky states": The proposition states that a tricky states exists if the number of empty slots is less than 2(H-1). However, at lines 38 and 39, the authors are mentioning that what can cause the existence of a tricky state is the inability to reshuffle unfixed containers. In other words, could it happen that, despite E < 2(H - 1) being true, a tricky states does not exist (for example, in the case in which the number of unfixed containers in the remaining two stacks is less than E)? It seems to me that the existence of a tricky state must depend on the number of unfixed containers in the “extreme" state. Please, clarify this doubt.

We didn’t present our idea clearly in the old version. The reviewer’s thought is right, the existence of a tricky state (now called dead-end state) depends on ***f***. Proposition 2 wants to justify the possibility of entering dead-end state of a layout before the algorithm actually starts.

We have restated Proposition 2: “No dead-end state can be generated whatever movements and fixed vectors are applied to cases with E ≥ 2(H − 1). For cases with E < 2(H − 1), there exist possibilities that the algorithm falls into a dead-end state.”

Given a layout L, the algorithm proposed in our paper starts from fixed vector f=U•**1**, and fixes containers one by one (changes f in this process). For a layout L with E<2(H-1), it has possibilities that the algorithm encounters a dead-end state; for a layout L with E ≥ 2(H − 1), there is no chance for the layout to enter a dead-end state in the process of the algorithm. The example is also given on Page 10.

1. Page 12, lines 49-52: The concept of “bottom tiers protection" is quite interesting. However, one might wonder what is the effect of different choices of parameters value, namely 2 and S. Would computational results and running times change significantly if these two parameters were modified? How?

We have added a subsection (Section 6.1) discussing the experiment on tier-protection indicator. It shows that the tier-protection indicator is effective, as the performance with the indicator is better than that without the indicator (the setting (0,N)). Moreover, the settings with intermediate parameters have the best performance, and tight and loose parameters have worse performance. If the parameters are too tight, the instances even do not have a solution.

1. The authors should present the running times of the proposed algorithm on both benchmark sets (if possible, along with the running time of the two benchmark heuristics).

The running times of the three heuristics (TGH, LGVFH, FBH) are added. Specially, TGH and LGVFH are implemented by us.

1. Please, provide details of the implementation: Machine configuration (speed, ram, OS, etc.), language and compiler used, etc.

We added the machine information in the first paragraph of Section 6 “Experiments were conducted on a PC with Intel Core i7 CPU clocked at 3.40 GHz with Windows 7 operating system. The code was written in Java.”

Minors

1. Page 15, line 6 : The accomplishment for an internal task→The accomplishment of an internal task?
2. Page 15, line 6: “tack" →”task"

We have amended typos of Point 1 and 2.

1. Why is the gap of Tables 2 and 3 measured with respect to the number of containers? Why is this gap preferred over a standard improvement measure (e.g., gap = (gash-literature)/literature). For example, for the first row of table 2, to give a measure of the goodness of the GASH approach, I would use:In this case, the *γ* value would indicate the net improvement, in percentage, (in this specific case, loss) of the proposed method with respect to the best result from the literature.

We have removed the original gap, and compare the performance of our algorithm with TGH and LGVFH. The formulation used is the one pointed out by the reviewer In the new version, GASH is renamed FBH, shorted for Feasibility-Based Heuristic.

Reviewer4

1. First, the presentation needs to be enhanced. The current manuscript has many loose ends, making it hard to follow. For example, page 7 stated that "containers in the reachable tiers can be permuted into any wanted arrangement", and in page 10, it reads "it is a natural fact that when S>=3, the reachable tiers can be permuted into any wanted placement." The condition "S>=3" was not mentioned when the property was first stated on page 7. Moreover, this property needs a careful proof, or at least some justification.

In the new version, we have removed the statement “containers in the reachable tiers can be permuted into any wanted arrangement”. The fact is for S>=3, the containers in reachable tiers can be permuted into any wanted arrangement. The conclusion is given in “Wang, N. (2014). Optimization Study on Container Operations in Maritime Transportation. PhD thesis, City University of Hong Kong, Hong Kong”. The proof is detailed given in this work (can be retrieved from City University Libarary->E-resources->E-theses->By title), and we are preparing a paper for this claim now. As the conclusion has no relationship with the contents in this paper, we remove this conclusion to avoid misunderstanding.

1. Also related to this issue, the three propositions given in the manuscript need proofs as well.

The proofs for the first two propositions are given in the new version. And the proposition 3 is removed from the new version.

1. Another example appears on page 12, section 6.2. The first line in this section mentioned a "container-stack pair", which seems to be a "task" to me. More careful and precise writing could help the reader understand the issue more easily.

We used container-stack pair to refer to a task, and this is not precise. A container-pair is a valid task only when it satisfies four conditions listed in Section 5.2. This is our fault.

As we consolidated Section 5-7, the content is moved to Page 13. We have revised the imprecise writing.

1. Similar situations occur when "solvable" seems to refer to the same concept as "feasible", and when the term "tricky state" pops out without definition. In general, the presentation makes the results appear far more complicated than they really are.

“Solvable” appeared in Proposition 3. Proposition 3 has no relationship with the main contents in this paper, hence we removed Proposition 3.

“Tricky states” is renamed “dead-end states” for better understanding. In the old version, we define “tricky states” in the text under Definition 1. Maybe, it is not apparent. In the new version, we define “dead-end” states in Definition 6. All the discussions about the “dead-end” states are after the definition.

1. The second concern refers to sections 6 and 7, where the techniques are developed and assembled to form the greedy and speedy heuristic, GASP. Five algorithms are presented in detail, but insights are insufficient. In other words, the manuscript described in detail what the algorithms do, but did not provide sufficient information to explain to the readers why they are designed that way. On top of all these, the manuscript should justify the basic idea of fixing the containers one by one. This idea seems reasonable to me, but it needs a careful justification. After all, the purpose of publication is to contribute to the body of knowledge, in addition to providing pseudo code for programmers to follow

The initial thought for us is to propose a framework and several techniques, and introduce them separately to demonstrate that every part can be used solely. Other scholars can use the framework combined with other techniques. The proposed techniques can also be used in other frameworks. Hence, Section 5 explains framework, Section 6 explains techniques and Section 7 explains the assembled heuristic.

But we find such a presentation is too hard to follow. Now we just explain the assembled heuristic, and the details of the framework and techniques are introduced after the heuristic in the same section (Section 5). Useful parts will still be adopted if some reader thinks they are effective in the future research.

For the idea behind fixing containers one by one, we explain it in Section 5.1: The advantage of fixing containers one by one lies in twofold: 1) movements are more target-oriented. Some works move a container each time, but they don’t have a clear aim. Hence, it is difficult to measure the benefit of moves and moreover, moves are blind: a container can be unnecessarily moved from stack a to b in a round and then moved from stack b to a after several rounds. 2) fixed containers are not allowed to be moved. But unfixed orderly containers are allowed to be moved. This can distinguish fixed and unfixed orderly containers. Even though both are orderly, the statuses of both kinds of containers are totally different. In works which does not distinguish fixed and unfixed containers, some containers which do not need moved (fixed containers in our context) are involved in the movements.

For the idea behind other techniques such as tier-protection, pre-extreme are added in the new version.