Computers and Operations Research 89 (2018) 127–139

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

New insights on the block relocation problem

Fabien Tricoire[a,*], Judith Scagnetti[b], Andreas Beham[b]

[a] Department of Business Administration, University of Vienna, Oskar-Morgenstern-Platz 1, Vienna 1090, Austria
[b] Heuristic and Evolutionary Algorithms Laboratory, University of Applied Sciences Upper Austria, Softwarepark 11, Hagenberg 4232, Austria

A R T I C L E I N F O

Article history:
Received 19 May 2017
Revised 12 July 2017
Accepted 15 August 2017
Available online 16 August 2017

Keywords:
Block relocation problem
Heuristics
Branch-and-bound

A B S T R A C T

This article presents new methods for the block relocation problem (BRP). Although much of the existing work focuses on the restricted BRP, we tackle the unrestricted BRP, which yields more opportunities for optimisation. Our contributions include fast heuristics able to tackle very large instances within seconds, fast metaheuristics that provide very competitive performance on benchmark data sets, as well as a new lower bound that generalises existing ones. We embed it in a branch-and-bound algorithm, then assess the influence of various factors on the efficiency of branch-and-bound algorithms for the BRP.

1. Introduction

In a world where container port traffic increases steadily (The World Bank, 2014), the optimisation of logistics at container terminals is more relevant than ever. Containers are typically stored in stacks between their arrival at the terminal and their departure when a vehicle picks them up. Similarly, at steel production factories, slabs are stored in stacks between their production and the time when they are picked up by a vehicle for further delivery. In both contexts, space is limited and the way items are stacked, be they containers or slabs, has an impact on productivity. In both cases it is possible to reorganise stacks with a crane in order to improve productivity. For this reason, optimisation problems related to loading, unloading and premarshalling of stacks in storage areas have been the subject of increasing attention over the last few years, as emphasised in a recent survey by Lehnfeld and Knust (2014). In this article we provide new insights on the block relocation problem (BRP), which is an unloading problem which we succinctly describe now.

A set of n items, usually called blocks or containers, are organised into W stacks and have to be retrieved in a certain order. It is usually considered that item 1 has to be retrieved first, then item 2, and so on until item n. An item may only be retrieved if it is on top of its stack. Otherwise, the items that block it (i.e. are above it) must be relocated to another stack first. A relocation involves taking an item from the top of a stack and putting it on top of another stack. Additionally, stacks may not exceed a certain height $H_{\max}$. The goal is to retrieve all items while minimising the total number of relocations required to do so.

It is possible to represent the BRP using a graph: to each configuration of the items in the bay, which we call a *state*, we associate a node. To each possible relocation from one state to another, we associate an arc with weight 1. To each possible retrieval of an item, also representing a transition from one state to another, we associate an arc with weight 0. Considering the set V of all nodes (possible states) and the set A of all arcs (possible transitions between those states), we obtain a directed graph $G = (V, A)$. The shortest path from the initial state (source) to the empty state (sink) then yields the optimal solution to the BRP. An example of this graph representation is given in Appendix A. However the size of V and A grows exponentially with the number of items and the number of stacks. It is therefore impractical to consider this whole graph explicitly. Any optimisation method for the BRP consists in finding a path from source to sink while exploring only a subset of the whole graph.

The contribution of this article is fourfold. First, we develop new heuristic operations to quickly build high quality solutions for the BRP. Second, we design a new constructive metaheuristic framework to use these operations in an even more efficient way, trading a bit of CPU effort for increased solution quality. Third, we develop a new lower bound for the BRP, which generalises previously known lower bound. Fourth, we conduct an extensive experimental study of all introduced mechanisms. This study allows us to define desirable features of branch-and-bound algorithms for the BRP, and shows that all developed heuristic components contribute to solving the BRP more efficiently.

The remainder of this article is organised as follows. In Section 2, we quickly survey recent literature on the BRP. In

* Corresponding author.
E-mail addresses: fabien.tricoire@univie.ac.at (F. Tricoire), judith.fechter@fh-hagenberg.at (J. Scagnetti), andreas.beham@fh-hagenberg.at (A. Beham).

http://dx.doi.org/10.1016/j.cor.2017.08.010
0305-0548/

Section 3 we present various heuristic methods for the unrestricted BRP. In Section 4 we introduce a new lower bound and present a branch-and-bound algorithm for the unrestricted BRP. Section 5 contains experimental validation for our contributed optimisation methods. Finally, we draw some conclusions and outline further research directions.

2. Literature review

The BRP has received notable attention in the last few years. In their recent survey, Lehnfeld and Knust (2014) present an overview of the scientific literature on loading, unloading and premarshalling problems. Using their terminology, the BRP is an unloading problem. They make a difference between *forced moves*, that involve relocating items blocking the next item to be removed, and *voluntary moves*, that involve any other type of relocation. Voluntary moves are also called *cleaning moves* by Petering and Hussein (2013). We want to emphasise this difference here since most of the existing contributions only consider forced moves. Put differently, the assumption is that the only items that may be relocated are those blocking the next item to be retrieved. From now on, we call this assumption A1, as named by Caserta et al. (2012). The BRP under assumption A1 is the restricted BRP, otherwise it is the unrestricted BRP. It should be noted here that under assumption A1, only two situations can occur: (i) Either the next item to be retrieved is on top of a stack, in which case it is retrieved; (ii) Or there are items blocking it, in which case these items are relocated. In that context, the only decision making that can be done is determining to which other stack these items should be relocated.

The BRP is first introduced by Kim and Hong (2006). They consider the now-common BRP setting where every item has a different priority, as well as a setting where priorities are given to groups of items. Their work is under assumption A1, i.e. only forced moves are considered. The authors propose a branch-and-bound algorithm to solve instances with up to 30 items and 6 stacks within less than one hour. They also design a heuristic rule to relocate items, that estimates the expected number of additional relocations needed after a certain relocation and greedily selects the relocation minimising that number. This heuristic produces solutions within one or two seconds and ranges on average between 2% and 16% from the optimum, depending on instance classes.

Lee and Lee (2010) present a three-phase heuristic for a BRP where the objective function also considers the distance between stacks. Assumption A1 is also made. In the first phase, a solution is constructed heuristically. In the second phase, the number of relocations is reduced by merging relocations of same items. This is performed using a mixed-integer program (MIP). In the third phase, a consideration of distance between stacks is added to a simplified version of the MIP from phase 2.

Caserta et al. (2011) also use assumption A1 and design a corridor method algorithm. The corridor method is a heuristic framework that explores limited amount of solutions within an exact framework, similar to what is done in beam search. In that case, the exact framework is a dynamic programming algorithm.

Jovanovic and Voss (2014) propose a chain heuristic that considers the next two items to be relocated and avoids to relocate the first one to the stack that is the best for the second one. This is under assumption A1. The authors insist on the difference between simple and complex methods. Simple methods typically compute within less than a second. They compare this method to other previous contributions, notably Wu and Ting (2010) and Caserta et al. (2012), and show that their chain heuristic performs better than the other simple heuristics. The best results reported are from the beam search of Wu and Ting (2010), but at the expense of CPU effort.

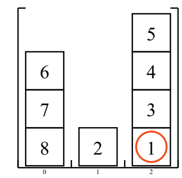

Fig. 1. For $H_{\max} = 4$, optimal solution with assumption A1 has 6 relocations but optimal solution without A1 has 4 relocations. Example taken from Caserta et al. (2012).

Wu and Ting (2010) develop a beam search algorithm for the BRP with assumption A1. The beam search is based on a breadth-first branch-and-bound algorithm. Using a depth-first scheme instead, the authors also propose a branch-and-bound algorithm. They also propose three heuristic rules to determine where the next item should be relocated.

Tanaka and Takii (2014) introduce a new lower bound for the restricted BRP. This lower bound is not applicable to the unrestricted BRP since it relies on the fact that items blocking the next item to remove have to be relocated next. Integrating it into a branch-and-bound algorithm, they show that their new lower bound improves the performance of branch-and-bound on the restricted BRP.

Caserta et al. (2012) provide a complexity study of the BRP. They prove that the BRP is NP-hard for any finite $H_{\max}$ and $W < n$. They then formulate two MIP models, BRP-I and BRP-II. BRP-I considers all possible relocations while BRP-II follows assumption A1. They also provide a simple example that emphasises that assumption A1 implies losing optimality. We reproduce this example in Fig. 1. However, only model BRP-II is tested, so there is no assessment of the cost of assumption A1. Finally, the authors also provide a simple heuristic under assumption A1.

Expósito-Izquierdo et al. (2015) correct the BRP-II model from Caserta et al. (2012), then present a new branch-and-bound algorithm for the restricted BRP. The branch-and-bound is compared to the A* algorithm from Expósito-Izquierdo et al. (2014) and results show that the newer method is faster.

Zehendner et al. (2015) present an improved mathematical formulation for the restricted BRP. They first correct the BRP-II model from Caserta et al. (2012), then provide an alternative model with less variables. This allows them to reduce CPU effort and to solve more instances than with the corrected BRP-II.

Forster and Bortfeldt (2012) develop methods for the unrestricted BRP where priorities are given to groups of items rather than to single items. They first develop an improved lower bound, compared to previous contributions simply considering the number of items that need to be relocated. They also develop a construction heuristic that applies what they call BG moves. BG stands for Bad-Good and involves the relocation of an item which was previously blocking another item into a stack where it does not block any item. If BG moves can be performed then they are performed, with a priority given to moves to a non-empty stack. A tree search is also presented. In order to keep CPU effort low, only certain moves are considered, so the method is similar to a beam search. BG moves are always preferred within this tree search procedure. The authors then compare their method to previous contributions, although these previous contributions are considering the restricted BRP.

Petering and Hussein (2013) consider the BRP without assumption A1. They first develop a MIP model, called BRP-III, with considerably less variables than BRP-I from Caserta et al. (2012). Although it provides a lower bound of worse quality than BRP-I, BRP-III still performs better on average. Still, the authors conclude that a mathematical programming approach is not sufficient for real-