A Robust model for the Restricted Block Relocation Problem

Tiziano Bacci[1], Sara Mattia[2], Paolo Ventura[3]

[1] Istituto di Analisi dei Sistemi ed Informatica - "Antonio Ruberti" del CNR, Via dei Taurini, 19 - 00185 Roma, Italy,
tiziano.bacci@iasi.cnr.it

[2] Istituto di Analisi dei Sistemi ed Informatica - "Antonio Ruberti" del CNR, Via dei Taurini, 19 - 00185 Roma, Italy,
sara.mattia@iasi.cnr.it

[3] Istituto di Analisi dei Sistemi ed Informatica - "Antonio Ruberti" del CNR, Via dei Taurini, 19 - 00185 Roma, Italy,
paolo.ventura@iasi.cnr.it

Abstract

We present a new robust variant of the Restricted Block Relocation Problem. We also propose a novel Integer Linear Programming formulation to solve the problem to exact optimality.

Keywords : *Block relocation problem, robust optimization, Integer Linear Programming.*

1 The Block Relocation Problem

The Block Relocation problem (BRP) is defined on a given set of blocks (items) $B = \{b_1, .., b_n\}$ located in the stacks $S = \{s_1, ..., s_m\}$ of a storage area (bay). Each stack has a given height (capacity) h and the blocks are stored in / retrieved from a stack according to a First-In Last-Out policy. The blocks of B must leave the bay in a prescribed order $\pi : \{1, ...n\} \to B$, where $\pi[i]$ is the i-th block to exit. When a block b_i leaves the bay, it must be retrieved from the stack s_j where it is currently located. Therefore, all the blocks located above b_i must be reshuffled (i.e. removed from s_j and reallocated in some other stacks of the bay). In the rest of the paper, for any given $a \in \mathbb{Z}_+$, we will use the notation $[a]$ for the set $\{1, ..., a\}$.

The Block Relocation Problem consists of minimizing the total number of block reshuffles needed to retrieve all the blocks of B from the bay. The problem arises from the management of a container bay, where a certain number of containers are piled into stacks. The retrieving operations are performed by cranes, which can only access the topmost container of each stack. Therefore, in order to retrieve a certain container, all the containers allocated on its top (*blocking containers*) must be reshuffled. These relocation operations are very expensive and their number must be minimized. The Block Relocation Problem has a crucial role in the logistics of containers and, since the number of containers shipped worldwide is dramatically growing, the problem has been widely investigated in the last years.

Here we consider the *restricted* BRP (RBRP), where only blocks above the next one to be retrieved can be reshuffled. Indeed, in order to simplify the process and reduce the number of possible movements in each step, the restricted policy is often used in the applications [8, 1]. In Figure 1 we give an example of a RBRP instance with 5 blocks initially located in a yard with 3 stacks s_1, s_2, s_3 of height 3. The retrieval order of the blocks is $(b_1, b_2, b_3, b_4, b_5)$. In the (optimal) solution illustrated in the picture, block b_4 is reshuffled from stack s_1 to stack s_3 in order to retrieve block b_1, and then to stack s_2 when block b_3 has to leave the yard. Hence, the total number of reshuffles (emphasized in gray in the picture) required by this solution is 2. Restricted BRP is known to be NP-hard and it is also very difficult to be solved in practice. For complexity results, refer to [2, 4, 5, 7]. For such problem, a sequence of exact

algorithms have been produced in the literature, that can be grouped into methods based on Integer Linear Programming (ILP) formulations and search-based methods. The most effective ILP-based method is the one recently proposed in [3]. The search-based algorithms include approaches that use different techniques to explore the solution space. Most of them implement a combinatorial Branch &Bound procedure. In particular, the algorithm proposed in [9] is definitively the most effective one in this second group.

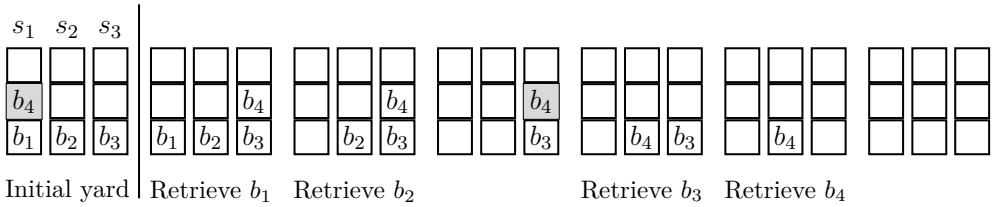

FIG. 1: An optimal solution to the Restricted Block Relocation Problem.

2 A robust variant of RBRP

In this paper, we investigate a possible *robust* model for the Restricted Block Relocation Problem, that we denote as *robRBRP*. In such a model we make the hypothesis that there exists a value $q \in [n]$ such that the first q items are retrieved following a given order $(\pi^0(1), ..., \pi^0(q))$, while for the remaining $n - q$ items, it is given set of d alternative retrieval orders $\pi^1, ..., \pi^d$, with $\pi^l : \{q + 1, ..., n\} \rightarrow B$, for each $l \in [d]$. Only one of these orders (scenarios), say $\pi^* \in \{\pi^1, ..., \pi^d\}$, will be operative at the end but this will be known only after the retrieval of the first q blocks. The goal here is to decide how to realize the reshuffle operations needed to retrieve the first q items in order to minimize the total number of reshuffles needed in the worst case scenario. This is similar to the model proposed in [10], where the retrieval order of the items is completely unknown at the beginning and it is revealed only over time. In that case, any retrieving order can represent a possible scenario and the authors propose an on-line heuristic strategy with a guaranteed competitive ratio. Instead, here only certain given scenarios must be considered. This can realistically represent many cases arising in practical applications where the order of the first items that needs to be retrieved can be given as fixed while the uncertainty comes from the fact that, after a certain time horizon, the arrival to the container yard of two or more given trains (or vessels, or trucks), each in charge to retrieve a certain set of items, can not be completely predicted. Therefore, in this scenario, only a few variants of an *ideal* order have to be considered.

2.1 Upper and lower bounds to the optimal solution of robRBRP

A possible approach to construct a feasible solution to the problem is the following two phases approach. First one constructs the optimal solution in order to retrieve items $\pi^0(1), ..., \pi^0(q)$ within the minimal number of reshuffles. Then, the other $n - q$ items are retrieved in the sequence $\pi^*(q + 1), ..., \pi^*(n)$, for the revealed order $\pi^* \in \{\pi^1, ..., \pi^d\}$. The total number of reshuffles performed in the first and in the second phase of the algorithm represents an upper bound to the optimal solution of robRBRP. The possible gap between the two values is due to the delayed knowledge of π^*. Indeed, consider the example in Figure 2. Here, 6 items are initially located in a yard with 3 stacks of height 5. Moreover, we have $q = 2$, $\pi^0(1) = b_1$, $\pi^0(2) = b_2$, and two alternative orders π^1 and π^2 with $\pi^1(3) = b_3, \pi^1(4) = b_4, \pi^1(5) = b_5, \pi^1(6) = b_6$ and $\pi^2(3) = b_4, \pi^2(4) = b_3, \pi^2(5) = b_5, \pi^2(6) = b_6$. In order to retrieve block b_1, we have to reshuffle b_3, b_6, b_5, and b_4. According to solution X, we can move all these blocks in stack s_3. Then also b_2 can exit the yard with no further reshuffles. Therefore, X represents the solution that minimizes the number of reshuffles (4) needed to retrieve the first q blocks. An alternative solution is the one defined by Y. In this case, b_6, b_5 and b_4 are initially moved to

stack s_3, while b_3 is allocated above b_2 in stack s_2. Then, in order to retrieve b_2, b_3 needs to be reshuffled again. Therefore, according Y, we need 5 reshuffles to retrieve b_1 and b_2. Now, once b_1 and b_2 have been retrieved we know which one of the two orders π^1 and π^2 does actually apply. It is not difficult to see that, starting from solution X, we need at least 4 (in case order π^1 is realized) or 2 (in case π^2 applies) more reshuffles. Instead, if we start from solution Y, we can retrieve, for both possible orders π^1 and π^2, all the remaining blocks with no further reshuffle.

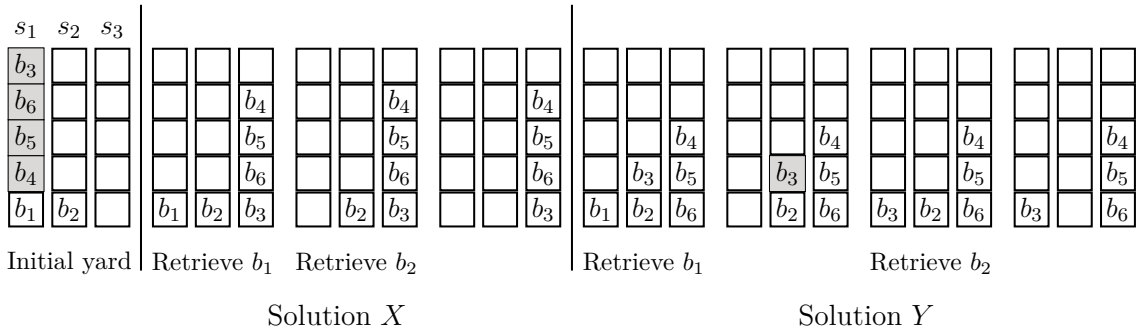

FIG. 2: Two possible solutions to retrieve items b_1 and b_2

Moreover, observe that the optimal solution of the RBRP problem defined as also the order π^* is known in advance is actually a lower bound for the optimal value of robRBRP. We compared the upper and lower bounds defined above on a data set of instances introduced in [6]. In particular, we solved the subproblems of the two phases heuristic algorithm as well as the RBRP problem that define the lower bound, by applying the exact procedure introduced in [3]. The results of the computational experiments are illustrated in Table 1. Here, each row represents a set of 40 instances of n blocks initially allocated in a yard with w stacks of height h. For each group of instances, first we considered 3 possible values for q: $\frac{n}{3}$, $\frac{n}{2}$, and $\frac{2n}{3}$ and then we randomly generated $d = n - q$ possible scenarios. In columns LB^{min}, LB^{mean}, and LB^{max} (UB^{min}, UB^{mean}, and UB^{max}) we report the mean (among all the instances of the group) of the minimum, mean and maximum value of the lower bound (upper bound, resp.) obtained among all the considered scenarios. As you can see, the average mean percentage gap among lower and upper bounds varies from about 17%, for the smallest instances, up to about 56%, for the largest ones.

(n,m,h)	$\frac{q}{n}$	d	LB^{min}	LB^{mean}	LB^{max}	UB^{min}	UB^{mean}	UB^{max}
(21,7,5)	$\frac{1}{3}$	14	7.0	9.2	11.5	14.5	16.7	18.9
(21,7,5)	$\frac{1}{2}$	11	7.8	9.2	10.7	16.4	18.3	20.0
(21,7,5)	$\frac{2}{3}$	7	8.5	9.2	10.0	16.8	17.9	19.2
(24,6,6)	$\frac{1}{3}$	16	11.3	13.9	16.7	20.9	23.9	27.2
(24,6,6)	$\frac{1}{2}$	12	12.6	14.2	15.9	25.2	27.4	29.5
(24,6,6)	$\frac{2}{3}$	8	13.2	14.0	14.8	27.7	29.1	30.3
(35,7,7)	$\frac{1}{3}$	24	21.0	24.7	28.5	41.4	45.6	50.0
(35,7,7)	$\frac{1}{2}$	18	22.3	24.6	27.2	52.4	55.5	58.5
(35,7,7)	$\frac{2}{3}$	12	23.3	24.6	25.9	54.4	56.4	58.3

TAB. 1: Upper and lower bound values for the robRBRP

2.2 An Integer Linear Programming formulation for robRBRP

Here we propose an ILP based exact approach to solve our robust variant of the problem. In particular, we derive a suitable ILP formulation (in the following denoted rob3BRP) for robRBRP, starting from the one presented in [3] for the restricted BRP (in the following, 3BRP). The 3BRP formulation makes use of two kind of binary variables, each defined for all

$i \in [n], j \in [w]$, and $t \in [i]$: $x_{i,j,t}$ (that is 1 if item $\pi(i)$ is located in stack j at time t) and $y_{i,j,t}$ (that is 1 if item $\pi(i)$ is reshuffled from stack j at time t). The rob3BRP formulation uses different copies of the variables x and y. In particular, for all $t \in \{1, ..., q\}$, we consider the variables $x^0_{i,j,t}$ and $y^0_{i,j,t}$. Then, for $t \in [q+1, ..., n]$ we will have instead s copies of such variables $(x^l_{i,j,t}, y^l_{i,j,t})$, each indexed by a different scenario $l \in [d]$.

The constraints of the rob3BRP formulation are derived from the ones of 3BRP as follows: i) each variable $x_{i,j,t}$ ($y_{i,j,t}$) defined on a time index $t \in [q]$, is replaced by the corresponding variable $x^0_{i,j,t}$ ($y^0_{i,j,t}$, resp.); ii) each constraint containing one ore more variables defined on a time period $t > q$ is duplicated d times and, in each copy, such variables are substituted by the ones indexed by the corresponding scenario. Then we have to match the values of the x^0 and y^0 variables with those of x^l and y^l at time q. This is done by adding the constraints $x^0_{i,j,q} = x^l_{i,j,q}$, and $y^0_{i,j,q} = y^l_{i,j,q}$, for all $i \in [n], j \in [w]$, and $l \in [d]$. Moreover, in order to minimize the number of reshuffle operations in the worst-case scenario, we need to introduce a dummy variable Z and the constraints $Z \geq \sum_{i\in[n]} \sum_{j\in[w]} \sum_{t\in\{q+1,...,n\}} y^l_{i,j,t}$, for all $l \in [d]$. Finally, the objective function is to minimize $\sum_{i\in[n]} \sum_{j\in[w]} \sum_{t\in[q]} y^0_{i,j,t} + Z$.

References

[1] T. Bacci, S. Conte, D. Matera, S. Mattia, and P. Ventura. A new software system for optimizing the operations at a container terminal. In *A View of Operations Research Applications in Italy, 2018*, volume 2 of *AIRO Springer Series*, pages 41–50. Springer, Cham, 2019.

[2] T. Bacci, S. Mattia, and P. Ventura. Some complexity results for the minimum blocking items problem. In *Optimization and Decision Science: Methodologies and Applications*, volume 217 of *Springer Proceedings in Mathematics & Statistics*, pages 475–483. Springer International Publishing, 2017.

[3] T. Bacci, S. Mattia, and P. Ventura. A branch and cut algorithm for the restricted block relocation problem. *European Journal of Operational Research*, (in press, DOI: https://doi.org/10.1016/j.ejor.2020.05.029), 2020.

[4] F. Bonomo, S. Mattia, and G. Oriolo. Bounded coloring of co-comparability graphs and the pickup and delivery tour combination problem. *Theoretical Computer Science*, 412(45):6261–6268, 2011.

[5] M. Caserta, S. Schwarze, and S. Voß. A mathematical formulation and complexity considerations for the blocks relocation problem. *European Journal of Operational Research*, 219(1):96–104, 2012.

[6] M. Caserta, S. Voß, and M. Sniedovich. Applying the corridor method to a blocks relocation problem. *OR Spectrum*, 33(4):915–929, 2011.

[7] K. Jansen. The mutual exclusion scheduling problem for permutation and comparability graphs. *Information and Computation*, 180(2):71–81, 2003.

[8] J. Lehnfeld and S. Knust. Loading, unloading and premarshalling of stacks in storage areas: Survey and classification. *European Journal of Operational Research*, 239(2):297–312, 2014.

[9] S. Tanaka and F. Mizuno. An exact algorithm for the unrestricted block relocation problem. *Computers & Operations Research*, 95:12–31, 2018.

[10] E. Zehendner, D. Feillet, and P. Jaillet. An algorithm with performance guarantee for the online container relocation problem. *European Journal of Operational Research*, 259(1):48 – 62, 2017.