European Journal of Operational Research xxx (xxxx) xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing, Transportation and Logistics

Reformulated acyclic partitioning for rail-rail containers transshipment

Maria Albareda-Sambola [a,*], Alfredo Marín [b], Antonio M. Rodríguez-Chía [c]

[a] Departament d'Estadística i Investigació Operativa, Universitat Politècnica de Catalunya. Barcelona Tech, Spain
[b] Departamento de Estadística e Investigación Operativa, Universidad de Murcia, Spain
[c] Departamento de Estadística e Investigación Operativa, Universidad de Cádiz, Spain

ARTICLE INFO

Article history:
Received 20 February 2018
Accepted 8 February 2019
Available online xxx

Keywords:
Combinatorial optimization
Rail-rail transshipment
Integer programming
Graph partitioning

ABSTRACT

Many rail terminals have loading areas that are properly equipped to move containers between trains. With the growing throughput of these terminals all the trains involved in a sequence of such movements may not fit in the loading area simultaneously, and storage areas are needed to place containers waiting for their destination train, although this storage increases the cost of the transshipment. This increases the complexity of the planning decisions concerning these activities, since now trains need to be packed in groups that fit in the loading area, in such a way that the number of containers moved to the storage area is minimized. Additionally, each train is only allowed to enter the loading area once.

Similarly to previous authors, we model this situation as an *acyclic graph partitioning problem* for which we present a new formulation, and several valid inequalities based on its theoretical properties. Our computational experiments show that the new formulation outperforms the previously existing ones, providing results that improve even on the best exact algorithm designed so far for this problem.

1. Introduction

Inland freight transportation is an increasing activity with a high economical and environmental impact. Among the alternative transport modes, road transport has traditionally had a very large share (for instance, according to Eurostat Statistics Explained, 2017 in 2014 it accounted for 74.9% of the total inland freight transport -t–Km- within the EU). In order to increase its competitiveness and to avoid the drawbacks of this activity such as congestion and pollution, other transportation modes are encouraged by the authorities (see, e.g., European Commission, 2011). In this regard, rail transport (accounting already in 2014 for almost 20% of the total inland freight transport in the European Union) is a relevant alternative, and presents now an increasing trend. This desirable increase in the rail freight transportation requires an adaptation of the infrastructures and involves increasingly complex decisions from the managerial point of view. In particular, the new challenges within the freight handling processes in railway yards have originated a stream of literature on suitable optimization approaches and decision support systems. A recent survey on these works can be found in Boysen, Fliedner, Jaehn, and Pesch (2013).

Among the surveyed problems, Boysen, Jaehn, and Pesch (2011) consider the rail-rail transshipment system described next. In a given railway yard, a series of containers must be transshipped among a number of trains. There is a *loading area* composed by a series of parallel train tracks equipped with gantry cranes, and a storage area. Each train has a given length and the tracks have a limited capacity, so that not all trains can be within the loading area at the same time, and a schedule must be designed where one group of trains enters the loading area after the other. Each time, containers are moved between trains within the loading area. If a container has to be moved from one of those trains to another one which is not there, it is placed in the storage area to wait until the train that must carry it enters the loading area. Feasible schedules are those associated with groups of trains that fit in the loading area, where no train needs to enter this area more than once. The goal is to find a feasible schedule that minimizes the number of containers that need to be moved to and from the storage area (see Fig. 1).

Already in Boysen et al. (2011) and later in Nossack and Pesch (2014a), this situation was addressed through a graph partitioning problem (GPP). In these problems, the nodes of a given graph must be assigned to clusters in order to optimize an objective function. Different GPP variants arise when alternative conditions are imposed on the number or the size of these clusters, or on the structure of the graph they induce. Some of them are surveyed in Alpert and Kahng (1995) and in the recent book in Parrochia and Neuville (2013). GPPs are rather difficult to solve through integer programming mainly due to the symmetries inherent to the

* Corresponding author.
E-mail address: maria.albareda@upc.edu (M. Albareda-Sambola).

https://doi.org/10.1016/j.ejor.2019.02.022
0377-2217/

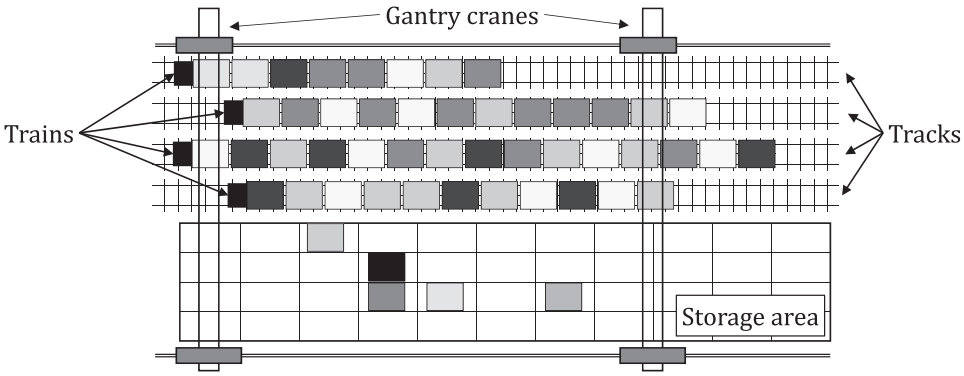

Fig. 1. Rail-Rail transshipment yard representation.

solutions. Relevant heuristics based on greedy strategies have been proposed for the case of bi-partitioning problems (e.g. Kernighan & Lin, 1970 and Fiduccia & Mattheyses, 1982). Their procedures start with a feasible solution and iteratively move to the best neighboring solution. Sanchis (1989) extended this idea to multi-way partitioning.

A GPP variant closely related to this paper is the clique partitioning problem, where the nodes of an undirected graph must be partitioned in order to maximize the total weight of the edges within each cluster. Three relevant papers on this problem are Grötschel and Wakabayashi (1989, 1990) and Oosten, Rutten, and Spieksma (2001). In Oosten et al. (2001), the structure of the associated polytope is studied and a cutting plane algorithm is developed. Branch-and-bound algorithms are given in Dorndorf and Pesch (1994) and Jaehn and Pesch (2013).

To model practical situations, additional constraints are often required. For instance, the *capacitated partitioning problem* (CPP) considers weights associated with the nodes and limits the total weight of the nodes in the same cluster. Also, the number of clusters may be fixed. Holm and Sørensen (1993) present a branch-and-cut algorithm for this case, where cuts are added to the formulation to avoid equivalent solutions. Branch-and-price methods based on different formulations and ways of solving the subproblems are implemented in Johnson, Mehrotra, and Nemhauser (1993) and Mehrotra and Trick (1998). Ferreira, Martin, de Souza, Weismantel, and Wolsey (1996) propose a multi-cut formulation for the CPP which is solved in Ferreira, Martin, de Souza, Weismantel, and Wolsey (1998) using branch-and-cut. Ji and Mitchell (2007) solve a clique partitioning problem with a minimum number of nodes in each clique with a branch-and-price-and-cut algorithm. Benati, Puerto, and Rodríguez-Chía (2017) also address a CPP with the additional constraint that the nodes in the same clique must be connected with respect to a secondary graph. Wong, Young, and Mak (2003) address the CPP with a heuristic based on clustering. Recent studies on graph partitioning problems are Bartolini, Casini, and Detti (2014) and Miyauchi and Sukegawa (2015).

The directed graph used to model the above application has one node associated with each train and arc (i, j) exists if and only if a container must be moved from train i to train j. Thus, the graph has weights associated with both, nodes and arcs. Nodes have to be partitioned into clusters with the objective of maximizing the total weight of the arcs within the clusters, but the total weight of the nodes in the same cluster is bounded above, and the digraph induced by the clusters cannot contain any circuit (otherwise, some train would have to enter the loading area more than once). The problem is consequently named *acyclic partitioning problem* (APP). This is an NP-hard problem even if all (node and arc) weights are equal to 1 (see Garey & Johnson, 1979). A heuristic method for the APP is given in Cong, Li, and Bagrodia (1994). To the best of our knowledge, the only exact methods for the APP are the pseudo-polynomial time algorithm by Lukes (1974) for the particular case defined on trees, and the works Nossack and Pesch (2014a,b) for the general case. The last works present the first integer programming formulation for the APP and develop an efficient branch-and-bound algorithm for it. The contribution of our paper is a new formulation able to solve much larger instances than before, based on the study of several properties of the problem solutions. The formulation is enforced with several families of valid inequalities. Hovering over all this study the idea of avoiding equivalent, symmetric solutions is always present.

The rest of the paper is organized as follows. In Section 2 we present in detail the problem we are considering in the paper. Several properties of this problem are highlighted in Section 3. Regarding Integer Programming formulations, Section 4 is devoted to the previously existing ones and 5 deals with the formulation proposed in this paper. Although the new formulation is the result of an elaborate process of refinement, still an exhaustive set of valid inequalities for it has been devised and presented in Section 6. We consider also an alternative, larger but tighter formulation in Section 7. A complete computational study (Section 8) and some conclusions (Section 9) close the paper.

2. The problem

In order to formally define the *Acyclic Partitioning Problem* (APP), we consider a directed graph (digraph) $G = (N, A)$ given by a set of nodes $N = \{1, \dots, n\}$ and a set of arcs $A \subseteq N \times N$. Without loss of generality we assume that $(i, i) \notin A \ \forall i \in N$. Associated with any node $i \in N$ there is a weight $w_i \geq 0$, and associated with any arc $(i, j) \in A$ there is a benefit $c_{ij} \geq 0$.

A partition of N is given by p nonempty clusters $N_1, \dots, N_p \subseteq N$ such that $\bigcup_{s=1}^{p} N_s = N$ and $N_s \cap N_t = \emptyset \ \forall s \neq t \in \{1, \dots, p\}$. Given $B > 0$, a partition $N_1, \dots, N_p$ is called B-feasible (or simply a B-partition) if $\sum_{i \in N_s} w_i \leq B \ \forall s \in \{1, \dots, p\}$. The value of partition $\mathscr{P} = \{N_1, \dots, N_p\}$ is defined as

$$v(\mathscr{P}) = \sum_{s=1}^{p} \sum_{(i,j) \in (N_s \times N_s) \cap A} c_{ij}.$$

$\mathscr{P}$ *induces* a digraph with nodes $\mathscr{P}$ and arcs $\{(N_s, N_t) : s \neq t, (N_s \times N_t) \cap A \neq \emptyset\}$. To clearly distinguish G from this induced digraph, G will be referred to as the *initial* digraph.

A circuit (directed cycle or simply cycle) in G is given by a sequence of different nodes $(i_1, \dots, i_r)$ such that $(i_q, i_{q+1}) \in A \ \forall q \in \{1, \dots, r-1\}$ and $(i_r, i_1) \in A$. A digraph is acyclic if it does not contain any directed cycle, and is cyclic otherwise.

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

The APP is the problem of obtaining the B-feasible partition of N of maximum value among those whose induced digraph is acyclic.

We will assume throughout the remainder of this paper that all weights w_i and the constant B are integer valued.

3. Solution properties

In order to derive properties of the feasible solutions to the APP, some additional definitions are required.

Two different nodes $i, j \in N$ are said to be *connected* if a sequence $(i_1 = i, i_2, \ldots, i_r = j)$ exists such that $\{(i_q, i_{q+1}), (i_{q+1}, i_q)\} \cap A \neq \emptyset \ \forall q \in \{1, \ldots, r-1\}$. Connectivity is an equivalence relationship whose classes are called *connected components of G*. When the number of its connected components is 1, G is said to be *connected*.

Given the initial digraph (N, A), $i \in N$ is said to be a *predecessor* of $j \in N$ when there exists a sequence of nodes $(i_1 = i, i_2, \ldots, i_r = j)$ such that $(i_q, i_{q+1}) \in A \ \forall q \in \{1, \ldots, r-1\}$. It is also said that j is a *successor* of i. To denote precedence relationships, we will use binary values α_{ij}, defined for $i, j \in N$, taking value 1 only if i is a predecessor of j. Note that these coefficients are easy to compute from the adjacency matrix of graph G by means of a greedy procedure.

Proposition 3.1. *If G is not connected and the APP is feasible, there exists an optimal solution to the APP where the nodes of different connected components are allocated to different clusters.*

Proof. Let $\mathscr{P} = \{N_1, \ldots, N_p\}$ be a feasible partition and assume that, for some $s \in \{1, \ldots, p\}$, N_s contains nodes from different connected components of G. Then, N_s can be split into $N_s = N_s^1 \cup N_s^2$, where N_s^1 contains all the nodes belonging to one of the connected components, and N_s^2 contains those belonging to the others. Consider the partition $\mathscr{P}' = (\mathscr{P} \setminus \{N_s\}) \cup \{N_s^1, N_s^2\}$. $\mathscr{P}'$ is clearly B-feasible since the sizes of the clusters in $\mathscr{P}'$ are bounded by those in $\mathscr{P}$.

Also, since the graph induced by $\mathscr{P}$ is acyclic, any cycle in the graph induced by $\mathscr{P}'$ should involve nodes N_s^1 and N_s^2, but these nodes cannot be connected in the graph induced by $\mathscr{P}'$ since sets N_s^1 and N_s^2 contain nodes belonging to different connected components of the initial graph. Consequently, the graph induced by $\mathscr{P}'$ is also acyclic.

The value of the partition $\mathscr{P}'$ is exactly the same as the value of $\mathscr{P}$, since no arcs exist connecting the two new clusters. Consequently, $v(\mathscr{P}') = v(\mathscr{P})$ and $\mathscr{P}'$ is also optimal.

Iterating in this way until all nodes of different connected components are allocated to different clusters, a new partition with this property and the same value is obtained. □

Note that, according to this property, an optimal solution to the APP for a general graph can be efficiently found by combining the optimal partitions of its connected components. Taking advantage of this fact, and without loss of generality, from now on we will assume that the initial graph G is connected.

Observe now that a node $i \in N$ is at the same time a successor and a predecessor of another node $j \in N$, $j \neq i$, if and only if there exists a directed cycle in G containing i and j. Observe also that, in this case, a partition of N such that $i \in N_s$ and $j \in N_t$ with $s \neq t$ will necessarily induce a cyclic digraph. As a consequence, all nodes belonging to a directed cycle in (N, A) must be assigned to the same cluster at any feasible solution to the APP. This fact makes it advisable to preprocess the original graph, by shrinking each cycle into one single node, as it was already done in Nossack and Pesch (2014a). To be precise, given a cycle $(i_1, \ldots, i_r)$, replace nodes $\{i_1, \ldots, i_r\}$ with one single node, η, with weight $w_\eta = \sum_{s=1}^{r} w_{i_s}$. For each node $i \in N \setminus \{i_1, \ldots, i_r\}$, include the arc (i, η) (resp. (η, i)) if and only if for some $s \in \{1, \ldots, r\}$, $(i, i_s) \in A$ (resp. $(i_s, i) \in A$). The benefits of these arcs are computed as $c_{i\eta} = \sum_{s=1}^{r} c_{i,i_s}$ and $c_{\eta i} = \sum_{s=1}^{r} c_{i_s,i}$. From an optimal solution of the APP defined on the resulting graph, an optimal solution to the original problem instance can be built by including nodes $\{i_1, \ldots, i_r\}$ in the cluster to which η has been assigned. The value of this partition will be the value of the partition in the reduced graph plus the sum of the values of the arcs with both endpoints in $\{i_1, \ldots, i_r\}$.

This procedure can be repeated until no cycles are left in the graph, making it acyclic. Thus from now on we will assume that the initial graph G is both connected and acyclic. Under this assumption it is easy to check whether an instance of the APP is feasible. Indeed, a necessary and sufficient condition for feasibility in this case is: $w_i \leq B \ \forall i \in N$.

Example 3.1. To illustrate this transformation, we will use the example depicted in Fig. 2, where node numbers are given inside the circles representing them, node weights are given next to the nodes and arc benefits are given close to the arcs. Observe that the cycle (1,11,6) can be shrunk to a single node, η, with weight 5. In turn, $(\eta, 2)$ would also form a cycle that can be shrunk into a single node, a, with weight 7, and the cycle (3,5,12,8) can be shrunk into one node, b, with weight 9. In the right hand side of the figure, we display the resulting graph, where squares represent nodes obtained by shrinking one or more cycles. In this case, node a represents nodes {1, 2, 6, 11} of the original graph, and node b represents nodes {3, 5, 8, 12}. Benefit c_{7a}, for example, is obtained as the sum $c_{71} + c_{76} = 1 + 3 = 4$.

It is well known that a digraph is acyclic if and only if it admits a topological order, *i.e.*, a mapping $\sigma : N \longrightarrow N$ such that, for any $(i, j) \in A$, $\sigma(i) < \sigma(j)$. Such a mapping is shown for the reduced graph in Fig. 2, next to the nodes in boxed numbers. Indeed, by suitably renaming the graph nodes, we can assume that $i < j$ for any $(i, j) \in A$. This assumption will also be made in what follows. Moreover, since the graph induced by any feasible solution must also be acyclic, it must also admit a topological order. Based on this idea, our formulation will index the clusters defining a topological order on the induced graph, so as to guarantee that it is acyclic. These ideas can be further exploited using the concept of successors of a node as shown in the next propositions. Here, we denote with σ^{-1} the inverse mapping of mapping σ.

Proposition 3.2. *Let $i < j \in N$ such that $\alpha_{ij} = 1$. If nodes i and j are assigned to the same cluster in a feasible partition, this cluster must contain the whole set*

$$N_j^i := \{\ell \in \{i+1, \ldots, j-1\} : \alpha_{i\ell} \cdot \alpha_{\ell j} = 1\}.$$

Proof. Let $i < j \in N$ with $\alpha_{ij} = 1$ and $\ell \in N_j^i$, and consider a partition where i and j are assigned to cluster N_s and ℓ is assigned to cluster N_t with $t \neq s$. Then, since $\alpha_{i\ell} = 1$ and $\alpha_{\ell j} = 1$, N_t is both, a successor and a predecessor of N_s in the graph induced by the partition. Therefore, this partition is infeasible. □

We next prove another property that will help avoiding equivalent optimal APP solutions.

Proposition 3.3. *For any feasible instance of APP there exists an optimal partition $\mathscr{P}$ and a topological order $\sigma_{\mathscr{P}}$ on the digraph induced by $\mathscr{P}$ such that, for all $t \in \{1, \ldots, |\mathscr{P}| - 1\}$ with $\sigma_{\mathscr{P}}^{-1}(t) \neq \emptyset$, and $\sigma_{\mathscr{P}}^{-1}(t+1) \neq \emptyset$,*

$$\sum_{i \in \sigma_{\mathscr{P}}^{-1}(t)} w_i + \sum_{i \in \sigma_{\mathscr{P}}^{-1}(t+1)} w_i > B.$$

Proof. If the above condition does not hold for a given solution, it is possible to build a new solution by merging the two clusters $\sigma_{\mathscr{P}}^{-1}(t)$ and $\sigma_{\mathscr{P}}^{-1}(t+1)$. The cluster obtained doing so will be B-feasible, and no cycles will be formed. The value of the obtained solution is at least as large as the value of the original one. □

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

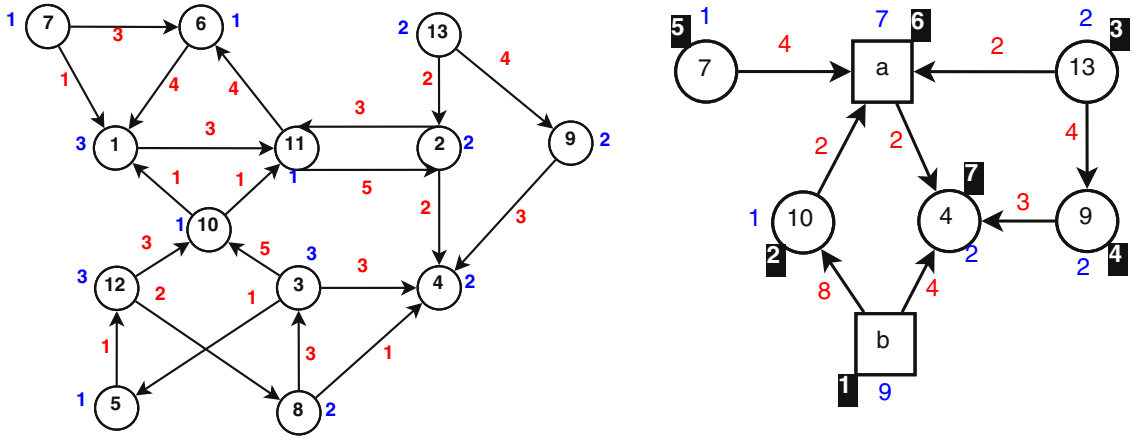

Fig. 2. Reduction of a cyclic graph.

Remark 3.1. Observe that the above proposition seems to contradict Proposition 3.1, in the sense that, Proposition 3.1 might suggest using more clusters than necessary and Proposition 3.3 tries to use the smallest possible number of clusters. However, this is not the case, since Proposition 3.3 assumes that the graph is connected and then Proposition 3.1 is not applicable.

4. Previous formulations

Nossack and Pesch (2014a) developed the first IP formulation for the APP. They considered auxiliary integer variables π_t, $t \in N$, together with the following families of binary variables:

$$x_{it} = \begin{cases} 1, & \text{if node } i \text{ belongs to cluster } N_t, \\ 0, & \text{otherwise,} \end{cases} \quad \forall i, t \in N,$$

$$z_{ijt} = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ belong to cluster } N_t, \\ 0, & \text{otherwise,} \end{cases} \quad \forall i < j \in N,\ t \in N,$$

$$y_{st} = \begin{cases} 1, & \text{if arc } (N_s, N_t) \text{ belongs to the induced digraph,} \\ 0, & \text{otherwise,} \end{cases} \quad \forall s \neq t \in N.$$

The model formulation, here named (NP), is given by

$$\text{(NP)} \quad \max \sum_{s \in N} \sum_{\substack{i,j \in N \\ i<j}} (c_{ij} + c_{ji}) z_{ijs}$$

$$\text{s.t.} \quad \sum_{s \in N} x_{is} = 1 \qquad \forall i \in N \qquad (1)$$

$$\sum_{i \in N} w_i x_{is} \leq B \qquad \forall s \in N \qquad (2)$$

$$z_{ijs} \leq x_{is} \qquad \forall s \in N, i < j \in N \qquad (3)$$

$$z_{ijs} \leq x_{js} \qquad \forall s \in N, i < j \in N \qquad (4)$$

$$x_{is} + x_{jt} - 1 \leq y_{st} \qquad \forall (i,j) \in A, s \neq t \in N \qquad (5)$$

$$\pi_s - \pi_t + n y_{st} \leq n - 1 \qquad \forall s \neq t \in N \qquad (6)$$

$$\sum_{i \in N} x_{is} \leq \sum_{i \in N} x_{i,s-1} \qquad \forall s \in \{2, \ldots, n\} \qquad (7)$$

$$2 z_{ihs} \leq z_{ijs} + z_{jhs} \qquad \forall s \in N, i < j < h \in N : \ j \in N_h^i \qquad (8)$$

$$z_{ihs} \leq z_{ijs} \qquad \forall s \in N, i < j < h \in N : \ j \in N_h^i \qquad (9)$$

$$z_{ijs} + z_{jhs} - z_{ihs} \leq 1 \qquad \forall s \in N, i < j < h \in N \qquad (10)$$

$$z_{ijs} - z_{jhs} + z_{ihs} \leq 1 \qquad \forall s \in N, i < j < h \in N \qquad (11)$$

$$-z_{ijs} + z_{jhs} + z_{ihs} \leq 1 \qquad \forall s \in N, i < j < h \in N \qquad (12)$$

$$x_{is} \in \{0,1\} \qquad \forall i, s \in N$$

$$y_{st} \in \{0,1\} \qquad \forall s \neq t \in N$$

$$\pi_s \in \mathbb{Z} \qquad \forall s \in N$$

$$z_{ijs} \in \{0,1\} \qquad \forall s \in N, \ i < j \in N.$$

Here, constraints (1) ensure that each node is assigned to a cluster, constraints (2) limit the weight of the clusters, constraints (3)–(4) ensure that two nodes belonging to different clusters do not contribute the objective, constraints (5) set the right arcs for the induced digraph. Circuits in the induced digraph are forbidden with constraints (6), a version of the Miller–Tucker–Zemlin (MTZ) constraints originally designed for the TSP (see Miller, Tucker, & Zemlin, 1960).

The remaining constraints are either valid inequalities or optimality constraints, added to the model to avoid symmetric solutions (it is the case of (7), that sort the clusters by their sizes) and/or improve the upper bound provided by the LP-relaxation of (NP). In particular, when nodes i and h are assigned to cluster s ($z_{ihs} = 1$) and another node j belongs to a directed path between i and h, (8) forces j to belong to N_s as well. When i and j are in different clusters, (9) prevents the successors of j from belonging to the same cluster as i. Finally, (10)–(12) are transitivity constraints (see Grötschel & Wakabayashi, 1990).

In a different paper, Nossack and Pesch (2014b) presented two other IP formulations for the APP. In the first one, the so-called *compact formulation*, they replaced the above z variables by the aggregated ones:

$$z_{ij} = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ belong to the same cluster,} \\ 0, & \text{otherwise,} \end{cases} \quad \forall i < j \in N.$$

The formulation is

$$\text{(NPC)} \quad \max \sum_{(i,j) \in A} c_{ij} z_{ij}$$

$$\text{s.t. (1), (2), (5), (6)}$$

$$z_{ij} + x_{is} - x_{js} \leq 1 \qquad \forall i < j \in N, \ s \in N \qquad (13)$$

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

$$2z_{ih} \le z_{ij} + z_{jh} \qquad \forall i < j < h \in N: \ j \in N_h^i \qquad (14)$$

$$z_{ih} \le z_{ij} \qquad \forall i < j < h \in N: \ j \in N_h^i \qquad (15)$$

$$z_{ij} + z_{jh} - z_{ih} \le 1 \qquad \forall i < j < h \in N \qquad (16)$$

$$z_{ij} - z_{jh} + z_{ih} \le 1 \qquad \forall i < j < h \in N \qquad (17)$$

$$-z_{ij} + z_{jh} + z_{ih} \le 1 \qquad \forall i < j < h \in N \qquad (18)$$

$$x_{is} \in \{0,1\} \qquad \forall i, s \in N$$

$$y_{st} \in \{0,1\} \qquad \forall s \ne t \in N$$

$$\pi_s \in \mathbb{Z} \qquad \forall s \in N$$

$$z_{ij} \in \{0,1\} \qquad \forall i, j \in N, \ i < j.$$

Similarly to (3) and (4), constraints (13) ensure that two nodes belonging to different clusters do not contribute the objective. Constraints (14)–(18) are valid inequalities analogous to (8)–(12).

The last formulation from Nossack and Pesch (2014b) considers the set $\mathscr{S}$ of clusters S satisfying the following three conditions: (i) the capacity constraint, (ii) if i, $h \in S$ then $N_h^i \subset S$, and (iii) $i \in S$, $j \notin S$ and $j \in N_h^i$ then $h \notin S$. Also, they defined $e^S = (e_1^S, \ldots, e_n^S)$ as an incidence vector of a cluster $S \in \mathscr{S}$, where $e_i^S = 1$ if node $i \in S$ and 0 otherwise. The formulation considers the auxiliary integer variables π_t, $t \in N$, for each $S \in \mathcal{S}$,

$$x_S = \begin{cases} 1, & \text{if cluster } S \text{ belongs to the chosen partition,} \\ 0, & \text{otherwise,} \end{cases}$$

and, for each $S_1 \ne S_2 \in \mathcal{S}, S_1 \cap S_2 = \emptyset$,

$$y_{S_1S_2} = \begin{cases} 1, & \text{if arc } (S_1, S_2) \text{ belongs to the induced digraph,} \\ 0, & \text{otherwise.} \end{cases}$$

The so-called *augmented set partitioning* formulation is the following.

$$\text{(NPA)} \ \max \sum_{S \in \mathscr{S}} c_S x_S \qquad (19)$$

$$\text{s.t.} \sum_{S \in \mathscr{S}} e_i^S x_S = 1 \qquad \forall i \in N \qquad (20)$$

$$e_i^{S_1} x_{S_1} + e_i^{S_2} x_{S_2} - 1 \le y_{S_1S_2} \qquad \forall (i,j) \in A, \ S_1 \ne S_2 \in \mathscr{S}, \ S_1 \cap S_2 = \emptyset \qquad (21)$$

$$\pi_{S_1} - \pi_{S_2} + n y_{S_1S_2} \le n - 1 \qquad \forall S_1 \ne S_2 \in \mathscr{S}, \ S_1 \cap S_2 = \emptyset \qquad (22)$$

$$x_S \in \{0,1\} \qquad \forall S \in \mathscr{S}$$

$$y_{S_1S_2} \in \{0,1\} \qquad \forall S_1 \ne S_2 \in \mathscr{S}, \ S_1 \cap S_2 = \emptyset$$

$$\pi_t \in \mathbb{Z} \qquad \forall t \in N,$$

where $c_S = \sum_{(i,j) \in A} c_{ij} e_i^S e_j^S \quad \forall S \in \mathscr{S}$. The objective function (19) maximizes the total benefit of the inter-cluster arcs, constraints (20) guarantee that each node belongs to one cluster, (21) force y and x variables to take consistent values. Again, MTZ constraints (22) forbid circuits in the induced digraph.

As already stated by the authors, the two last formulations are rather unpractical. Specifically, (NPC) has a weak LP-relaxation and exhibits awkward symmetries, and (NPA) contains many constraints and variables. As a consequence, Nossack and Pesch (2014b) does not present any computational study of these formulations.

5. New formulation

In this section, we introduce a new formulation for the APP. It uses solely x-variables as defined in formulation (NP) and the subset of the z-variables from (NPC) $\{z_{ij} : (i,j) \in A\}$ (note that this implies $i < j$). Additionally, for each pair $i < j \in N$ with $\alpha_{ij} = 1$, we define A_{ij} as:

$$A_{ij} = w_i + w_j + \sum_{\ell \in N_j^i} w_\ell.$$

Note that any cluster containing both, i and j, will have a total node weight of, at least, A_{ij}. The proposed formulation, denoted by (P), is

$$\text{(P)} \ \max \sum_{(i,j) \in A} c_{ij} z_{ij} \qquad (23)$$

$$\text{s.t.} \sum_{s \in K} x_{is} = 1 \qquad \forall i \in N \qquad (24)$$

$$\sum_{i \in N} w_i x_{is} \le B \qquad \forall s \in K \qquad (25)$$

$$\sum_{t \ge s} x_{it} + \sum_{t < s} x_{jt} \le 1 \qquad \forall i < j \in N: \ \alpha_{ij} = 1, \ A_{ij} \le B, \ \forall s \in K \qquad (26)$$

$$\sum_{t \ge s} x_{it} + \sum_{t \le s} x_{jt} \le 1 \qquad \forall i < j \in N: \ \alpha_{ij} = 1, \ A_{ij} > B, \ \forall s \in K \qquad (27)$$

$$z_{ij} + \sum_{t < s} x_{it} + \sum_{t \ge s} x_{jt} \le 2 \qquad \forall (i,j) \in A, \ \forall s \in K \qquad (28)$$

$$x_{is} \in \{0,1\} \qquad \forall i \in N, \ \forall s \in K$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in A.$$

Here, indices corresponding to the potential clusters are defined on the set $K = \{1, \ldots, k\}$, where k is an upper bound on the number of clusters in the optimal APP solution. In forthcoming sections we will discuss how to obtain its value. Observe that a possible value for k is n, but, obviously, by obtaining smaller k values we can reduce the size of formulation (P).

The objective function (23) measures the total benefit obtained from the arcs with both extremes in the same cluster. Each node must be assigned to one cluster, and this is guaranteed by constraints (24). The total weight of the nodes assigned to any cluster cannot exceed the bound B, as stated by constraints (25). The induced digraph cannot contain circuits, which is achieved by topologically ordering the clusters. Therefore, if $\alpha_{ij} = 1$ for $i, j \in N$, node i cannot belong to a cluster with a higher index than the cluster of node j. That is, for a given s value, if node j belongs to one of the first $s-1$ clusters, i cannot belong to cluster s nor to one with a higher index. If this happens the left hand side of the corresponding constraint (26) will take value 2. When i and j cannot be allocated to the same cluster because $A_{ij} > B$, index s can be also included in the second summation as stated by constraints (27).

In order to guarantee that the z-variables in the objective function reflect the allocation structure given by the x-variables, we use constraints (28). Here, given $(i,j) \in A$, if j is allocated to a cluster in position s or later, and i is allocated to a cluster in a position before s, they cannot be in the same cluster. The corresponding constraint (28) sets z_{ij} to zero.

Observe that, given the shape of the objective function and constraints (28), the integrality constraints on the z-variables can be relaxed. Indeed, since the problem consists in maximizing the sum of the benefits $c_{ij} (\ge 0)$ times the corresponding z-variables, then z-variables will take the largest possible value. In addition, using

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

(28), if i and j are in the same cluster $\sum_{t<s} x_{it} + \sum_{t\geq s} x_{jt} = 1$ for all $s \in K$, then z_{ij} will take value 1; otherwise, we can find $s_0 \in K$ such that $\sum_{t<s_0} x_{it} + \sum_{t\geq s_0} x_{jt} = 2$ and then $z_{ij} = 0$.

6. Formulation improvements

6.1. Preprocessing

Formulation (P) can be preprocessed by fixing some z-variables and x-variables to 0.

First, by Proposition 3.2 we can forbid beforehand that some pairs of nodes belong to the same cluster. Using the previously computed values A_{ij} we can fix

$$z_{ij} = 0 \qquad \forall (i,j) \in A : A_{ij} > B. \tag{29}$$

Second, consider a node i and the set of its successors, S_i. Considering the total weight of these successors plus the weight of i, $\hat{w} = w_i + \sum_{j\in S_i} w_j$, it is clear that there is a number of clusters, at the end of the topologically ordered list, where i cannot be allocated. Indeed, if i were assigned to cluster s and the total capacity of clusters $s, \ldots, k$ (which is equal to $(k-s+1)B$) is less than $\hat{w}$, the remaining nodes could not be correctly allocated. A similar reasoning can be made for the set of predecessors of i, P_i. Accordingly,

$$x_{is} = 0 \qquad \forall i \in N,\ s \in K :\ (k-s+1)B < w_i + \sum_{j\in S_i} w_j, \tag{30}$$

$$x_{is} = 0 \qquad \forall i \in N,\ s \in K :\ sB < w_i + \sum_{j\in P_i} w_j. \tag{31}$$

6.2. Valid inequalities

In this section we derive several valid inequalities for formulation (P). First we consider those previously used in the literature, i.e., in formulation (NPC).

Transitivity constraints (16), (17) and (18) can be incorporated to formulation (P) but need to be adapted in order to include only well-defined variables, i.e., $\forall i < j < \ell \in N :\ (i,j),(i,\ell),(j,\ell) \in A,\ A_{ij}, A_{i\ell}, A_{j\ell} \leq B$:

$$\left.\begin{array}{l} z_{i\ell} \geq z_{ij} + z_{j\ell} - 1 \\ z_{ij} \geq z_{i\ell} + z_{j\ell} - 1 \\ z_{j\ell} \geq z_{ij} + z_{i\ell} - 1 \end{array}\right\} \tag{32}$$

The meaning of these constraints is clear: Whenever a node i shares a cluster with two other nodes, j and ℓ, the latter two nodes must also share that cluster.

According to the variable definitions and Proposition 3.2, the following inequalities are also valid:

$$\left.\begin{array}{ll} z_{i\ell} \leq z_{ij} & \forall i < j < \ell \in N :\ (i,j),(i,\ell) \in A,\ A_{ij}, A_{i\ell} \leq B,\ j \in N_\ell^i \\ z_{i\ell} \leq z_{j\ell} & \forall i < j < \ell \in N :\ (i,\ell),(j,\ell) \in A,\ A_{i\ell}, A_{j\ell} \leq B,\ j \in N_\ell^i \end{array}\right\}. \tag{33}$$

Note that (33) are tighter than the related constraints (14) and (15), since (14) is obtained as the sum of both constraints in (33). The ideas behind these sets of constraints were also used for formulation (NPC), although they were not stated exactly in the same way. To the best of our knowledge, the valid inequalities that follow are presented in this work for the first time.

When three nodes, $i<j<\ell$, are allocated to the same cluster, any other node in the directed paths between them must belong to this cluster. Then, the total weight of that cluster is bounded below by

$$A'_{ij\ell} = w_i + w_j + w_\ell + \sum_{h\in N_j^i \cup N_\ell^j \cup N_\ell^i \setminus \{j\}} w_h.$$

So, if $A'_{ij\ell} > B$, the three nodes cannot share a cluster. Taking into account the transitivity, at most one of the variables z_{ij}, $z_{j\ell}$ and $z_{i\ell}$ can take value one. Therefore, for all $i<j<\ell \in N$ such that $A'_{ij\ell} > B$, the following are valid inequalities for (P):

$$\left.\begin{array}{ll} z_{ij} + z_{j\ell} + z_{i\ell} \leq 1 & (i,j),(i,\ell),(j,\ell) \in A,\ A_{ij}, A_{i\ell}, A_{j\ell} \leq B \\ z_{ij} + z_{i\ell} \leq 1 & (i,j),(i,\ell) \in A,\ A_{ij}, A_{i\ell} \leq B,\ \text{but } (j,\ell) \notin A \text{ and/or } A_{j\ell} > B \\ z_{i\ell} + z_{j\ell} \leq 1 & (i,\ell),(j,\ell) \in A,\ A_{i\ell}, A_{j\ell} \leq B,\ \text{but } (i,j) \notin A \text{ and/or } A_{ij} > B \\ z_{ij} + z_{j\ell} \leq 1 & (i,j),(j,\ell) \in A,\ A_{ij}, A_{j\ell} \leq B,\ \text{but } (i,\ell) \notin A \text{ and/or } A_{i\ell} > B. \end{array}\right\} \tag{34}$$

Capacity constraints (25) depend only on the x-variables. They cannot be stated using z-variables since z_{ij} is not defined if $(i,j) \notin A$. Nevertheless, valid inequalities in the same spirit are:

$$\left.\begin{array}{l} w_j + (w_i + \sum_{h\in N_j^i} w_h) z_{ij} + (w_\ell + \sum_{h\in N_\ell^j} w_h) z_{j\ell} + \sum_{\substack{h=1 \\ (h,j)\in A}}^{i-1} w_h z_{hj} \\ + \sum_{\substack{h=\ell+1 \\ (j,h)\in A}}^{n} w_h z_{jh} + \sum_{\substack{h=i+1 \\ (h,j)\in A \\ \alpha_{ih}=0}}^{j-1} w_h z_{hj} + \sum_{\substack{h=j+1 \\ (j,h)\in A \\ \alpha_{h\ell}=0}}^{\ell-1} w_h z_{jh} \leq B \\ \forall i < j < \ell \in N :\ (i,j),(j,\ell) \in A. \end{array}\right\} \tag{35}$$

Given (i,j), $(j,\ell) \in A$, the left hand side of the above inequality provides a lower bound on the sum of the weights of the nodes included in the same cluster as j. Indeed, the first term is the weight of node j, the second (third) addend accounts for the weights of the nodes in any path from i to j (from j to ℓ), excluding j because this weight is already included in the first term, whenever i and j (j and ℓ) are in the same cluster (Proposition 3.2). The fourth (fifth) addend is the sum of weights of the nodes, such that,: (i) have indices smaller than i (bigger than ℓ), (ii) are in the same cluster as j, and (iii) are linked by an arc with j (this third condition guarantees that these nodes have not been already included in the previous two addends because we are assuming the graph is topologically ordered). The sixth (seventh) addend gives the sum of weights of the nodes belonging to same cluster as j with indices between i and j (j and ℓ), linked to j by an arc but not belonging to any path between i and j (j and ℓ); this guarantees that these nodes have not been already included in the previous addends.

Taking advantage of the fact that if two extreme points of a path belong to the same cluster, then the full path is also contained in that cluster, additional capacity constraints can be stated as follows:

$$w_i + \sum_{\substack{j=1: \\ (j,i)\in A}}^{i-1} \left(\sum_{\ell\in L_j} w_\ell\right) z_{ji} + \sum_{\substack{j=i+1: \\ (i,j)\in A}}^{n} \left(\sum_{\ell\in R_j} w_\ell\right) z_{ij} \leq B \tag{36}$$

for all $i \in N$, any family of sets L_j satisfying

$$L_j \subseteq \{j\} \cup N_i^j, \quad L_{j_1} \cap L_{j_2} = \emptyset\ \forall j_1 \neq j_2, \quad \bigcup_{\substack{j=1: \\ (j,i)\in A}}^{i-1} L_j = P_i$$

and any family of sets R_j satisfying

$$R_j \subseteq \{j\} \cup N_j^i, \quad R_{j_1} \cap R_{j_2} = \emptyset\ \forall j_1 \neq j_2, \quad \bigcup_{\substack{j=i+1: \\ (i,j)\in A}}^{n} R_j = S_i.$$

Note that there can be empty sets L_j and/or R_j.

To separate this family of constraints we proceed as follows. Let $(\bar{x}, \bar{z})$ be the optimal solution to the linear problem. Then, for each $i \in N$, let

$$\bar{L}_j = \{\ell \in \{j\} \cup N_i^j :\ \bar{z}_{ji} = \max_p \{\bar{z}_{pi} :\ p \in P_\ell\}\}$$

where ties are broken in lexicographical order. Similarly, let

$$\bar{R}_j = \{\ell \in \{j\} \cup N_j^i :\ \bar{z}_{ij} = \max_p \{\bar{z}_{ip} :\ p \in S_\ell\}\}.$$

Given a tolerance T, if

$$w_i + \sum_{\substack{j=1: \\ (j,i)\in A}}^{i-1} \left(\sum_{\ell\in \bar{L}_j} w_\ell\right) \bar{z}_{ji} + \sum_{\substack{j=i+1: \\ (i,j)\in A}}^{n} \left(\sum_{\ell\in \bar{R}_j} w_\ell\right) \bar{z}_{ij} \geq B + T,$$

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

then the corresponding violated inequality is added to the formulation.

Constraints (28) ensure that z-variables take value 0 when required. These constraints can be extended to the following family of valid inequalities:

$$z_{ij} + z_{i\ell} + \sum_{t\geq s}(x_{jt} + x_{\ell t}) + \sum_{t<s} x_{it} \leq 3$$
$$\forall i < j < \ell \in N: \ (i,j),(i,\ell) \in A, \ A_{ij}, A_{i\ell} \leq B, \ \alpha_{j\ell} = 0,$$
$$A'_{ij\ell} > B, \ \forall s \in K. \tag{37}$$

When $(i,j) \in A$, constraints (26) can be extended to the following set of valid inequalities:

$$\sum_{t\geq s} x_{it} + \sum_{t\leq s} x_{jt} \leq 1 + z_{ij} \qquad \forall (i,j) \in A, \ A_{ij} \leq B, \ \forall s \in K. \tag{38}$$

Recall that formulation (P) uses an upper bound k on the number of clusters of the optimal partition. Therefore, the number of actual (nonempty) clusters in the obtained solution might be smaller than k. In this case, formulation (P) would leave some empty dummy clusters. These empty clusters introduce an awkward symmetry, since, according to (P), they can occupy arbitrary positions in the list $\{1,\ldots,k\}$. In order to break those symmetries, the following additional constraints can be added to force that the actual clusters have the lowest indices, and the empty ones, if any, are grouped at the end of the list:

$$\sum_{i\in N} w_i x_{it} \leq B \sum_{i\in N} x_{is} \qquad \forall s < t \leq k, \tag{39}$$

$$\sum_{i\in N}\sum_{t>s} x_{it} \leq (n-s)\sum_{i\in N} x_{is} \qquad \forall s < k. \tag{40}$$

Constraints (39) are based on node weights. If cluster s is empty, the right-hand side of (39) will be 0, forcing the clusters with greater indices to be empty as well; otherwise B is used to bound the right-hand side. Instead, (40) are based on the number of nodes. A zero value in the sum of the right-hand side will push all variables in the left-hand side, which correspond to clusters with a larger index, down to 0. In this case the left hand side contains more variables and the upper bound when the right-hand side is not zero is given by the maximum number of nodes that can be allocated to the last $n-s$ clusters. In the worst case this number is $n-s$.

The following valid inequalities also follow from Proposition 3.3.

$$\sum_{\substack{i\in N\setminus\{j\}\\ \alpha_{ji}=0}} w_i(x_{is} + x_{i,s+1}) \geq (B+1)\sum_{t=s+2}^{k} x_{jt} \quad \forall s \leq k-2, \ \forall j \in N, \tag{41}$$

$$\sum_{i<j} w_i x_{is} + \sum_{\substack{i>j\\ \alpha_{ji}=0}} w_i x_{is} + \sum_{i\in N} w_i x_{i,s+1} \geq (B+1)\sum_{t>s} x_{jt}$$
$$\forall s < k, \ \forall j \in N. \tag{42}$$

We next explain the rationale behind valid inequalities (41); similar arguments can be used to show that inequalities (42) are valid. The idea behind (41) is the following: given $s \leq k-2$ if there is a nonempty cluster with index greater than $s+1$ (left hand side of (41) equal to $(B+1)$), then the sum of the weights associated with the nodes in cluster s and $s+1$ should be at least $B+1$, otherwise it would contradict Proposition 3.3.

An additional family of valid inequalities is

$$\sum_{\substack{i\in N:\\ w_i > B/2}} x_{is} \leq 1 \qquad \forall s \in K. \tag{43}$$

6.3. Bounding the number of clusters

Observe that if the maximum number of clusters to be used (k) is small, the dimension of formulation (P) is drastically reduced. Therefore, the problem will be solved much easily if a small value of k is available, instead of the trivial bound on the number of clusters, n. However, it would be a wrong approach to solve the APP using any value k and then accepting the optimal solution in case of feasibility. Although this approach is correctly adopted for other problems with similar clusters structure, like the search of the chromatic number of a graph (Nobibon, Hurkens, Leus, and Spieksma, 2010, 2012), in our case feasibility does not guarantee optimality, as we show in the example illustrated in Fig. 3 where $w_i = 1 \ \forall i$, $B = 3$ and the benefits of the arcs are depicted next to them. Taking a value of $k = 8$, the optimal solution to this instance is given in Fig. 4 with optimal value 15.6. This solution only uses 6 out of the 8 possible clusters. Nevertheless, the actual optimal APP solution, with value 16, requires 9 clusters (see Fig. 5). The result given in Proposition 6.1 has been used to fix the value of k in our formulations:

Proposition 6.1. *An optimal partition $\mathscr{P}$ to APP exists, satisfying*

$$|\mathscr{P}| \leq \min\left\{n, 2\left\lfloor \frac{\sum_{i\in N} w_i}{B+1}\right\rfloor + \left\lceil \frac{\sum_{i\in N} w_i - (B+1)\left\lfloor \frac{\sum_{i\in N} w_i}{B+1}\right\rfloor}{B}\right\rceil\right\}. \tag{44}$$

Proof. It suffices to show

$$|\mathscr{P}| \leq 2\left\lfloor \frac{\sum_{i\in N} w_i}{B+1}\right\rfloor + \left\lceil \frac{\sum_{i\in N} w_i - (B+1)\left\lfloor \frac{\sum_{i\in N} w_i}{B+1}\right\rfloor}{B}\right\rceil. \tag{45}$$

By Proposition 3.3, the total weight of any pair of non-empty consecutive clusters can be assumed to be at least equal to $B+1$ (taking into account that all weights and B are integer numbers). Assuming that this lower bound is tight, the first addend of (45) keeps track of the filled couples of clusters. The numerator of the second addend stands for the remaining weight. If this rest, which is less than or equal to B, is not zero, an additional cluster must be added to the account. □

7. Alternative formulation

It became evident during the development of valid inequalities that those based on z-variables could be reinforced by including in the formulation additional z_{ij}-variables with $i<j \in N$ and $(i,j) \notin A$. At the expenses of having a larger formulation, there is a chance of obtaining better upper bounds, since the new constraints will be tighter on the z-variables in the objective function (23). To check if this alternative gives rise to better computational results, a new formulation (P') is considered:

$$\text{(P')} \ \max \sum_{(i,j)\in A} c_{ij} z_{ij} \tag{46}$$
s.t. (24),(25),(26),(27)
$$z_{ij} + \sum_{t<s} x_{it} + \sum_{t\geq s} x_{jt} \leq 2 \qquad \forall i < j \in N, \ \forall s \in K$$
$$x_{is} \in \{0,1\} \qquad \forall i \in N, \ \forall s \in K$$
$$z_{ij} \in \{0,1\} \qquad \forall i < j \in N. \tag{47}$$

The preprocessing phase developed in Section 6.1 is still valid for this formulation. In addition, similarly to formulation (P), the binarity constraints on the z-variables could be dropped from (P'). From the range of valid inequalities developed for formulation (P), those containing only x-variables remain the same for (P'): (39)–(43).

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

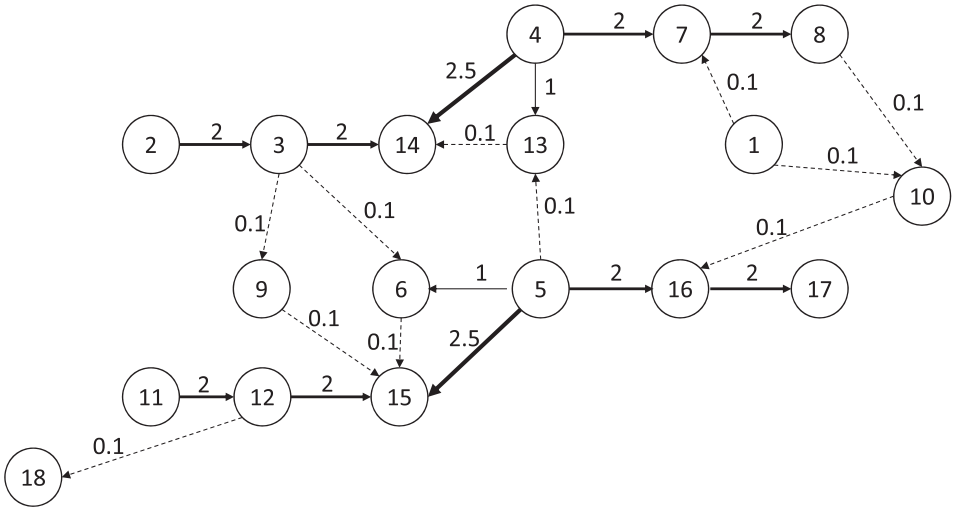

Fig. 3. Instance for counterexample.

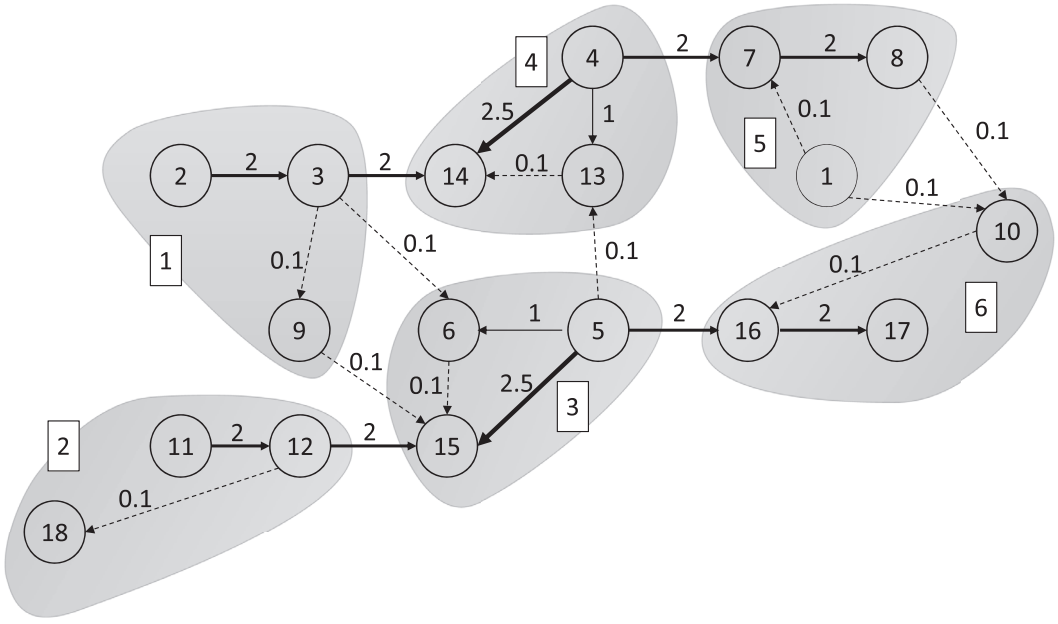

Fig. 4. Solution of the instance from Fig. 3 with $k = 8$.

Some other valid inequalities need to be modified in order to include the additional z-variables in (P'). Constraints (32), (33) and (34) are replaced, respectively, by:

$$\left.\begin{array}{l} z_{i\ell} \geq z_{ij} + z_{j\ell} - 1 \\ z_{ij} \geq z_{i\ell} + z_{j\ell} - 1 \\ z_{j\ell} \geq z_{ij} + z_{i\ell} - 1 \end{array}\right\} \quad \forall i < j < \ell \in N: \ A_{ij}, A_{i\ell}, A_{j\ell} \leq B, \tag{48}$$

$$\left.\begin{array}{l} z_{i\ell} \leq z_{ij} \quad \forall i < j < \ell \in N: \ A_{ij}, A_{i\ell} \leq B, \ j \in N^i_\ell \\ z_{i\ell} \leq z_{j\ell} \quad \forall i < j < \ell \in N: \ A_{i\ell}, A_{j\ell} \leq B, \ j \in N^i_\ell, \end{array}\right\} \tag{49}$$

$$\left.\begin{array}{ll} z_{ij}+z_{j\ell}+z_{i\ell} \leq 1 & \forall i < j < \ell \in N: \ A_{ij}, A_{i\ell}, A_{j\ell} \leq B, \ A'_{ij\ell} > B, \\ z_{ij}+z_{i\ell} \leq 1 & \forall i < j < \ell \in N: \ A_{ij}, A_{i\ell} \leq B \text{ and } A_{j\ell} > B \\ z_{i\ell}+z_{j\ell} \leq 1 & \forall i < j < \ell \in N: \ A_{i\ell}, A_{j\ell} \leq B \text{ and } A_{ij} > B \\ z_{ij}+z_{j\ell} \leq 1 & \forall i < j < \ell \in N: \ A_{ij}, A_{j\ell} \leq B \text{ and } A_{i\ell} > B, \end{array}\right\}. \tag{50}$$

Note that constraints (35) and (36) used the capacity of the clusters to limit the values of the z-variables. Formulation (P') allows to extend and simplify these constraints in the following way:

$$w_i + \sum_{j=1}^{i-1} w_j z_{ji} + \sum_{j=i+1}^{n} w_j z_{ij} \leq B \qquad \forall i \in N. \tag{51}$$

In addition, constraints (37) and (38) of (P) are replaced by the following ones:

$$z_{ij} + z_{i\ell} + \sum_{t \geq s} (x_{jt} + x_{\ell t}) + \sum_{t < s} x_{it} \leq 3$$
$$\forall i < j < \ell \in N: \ \alpha_{j\ell} = 0, \ A_{ij}, A_{i\ell} \leq B, \ A'_{ij\ell} > B, \ \forall s \in K, \tag{52}$$

$$\sum_{t \geq s} x_{it} + \sum_{t \leq s} x_{jt} \leq 1 + z_{ij} \qquad \forall i < j \in N, \ A_{ij} \leq B, \ \forall s \in K. \tag{53}$$

8. Computational study

In order to evaluate the performance of the new formulations and valid inequalities, a computational study has been carried out using the unique data set in the literature, presented in Nossack

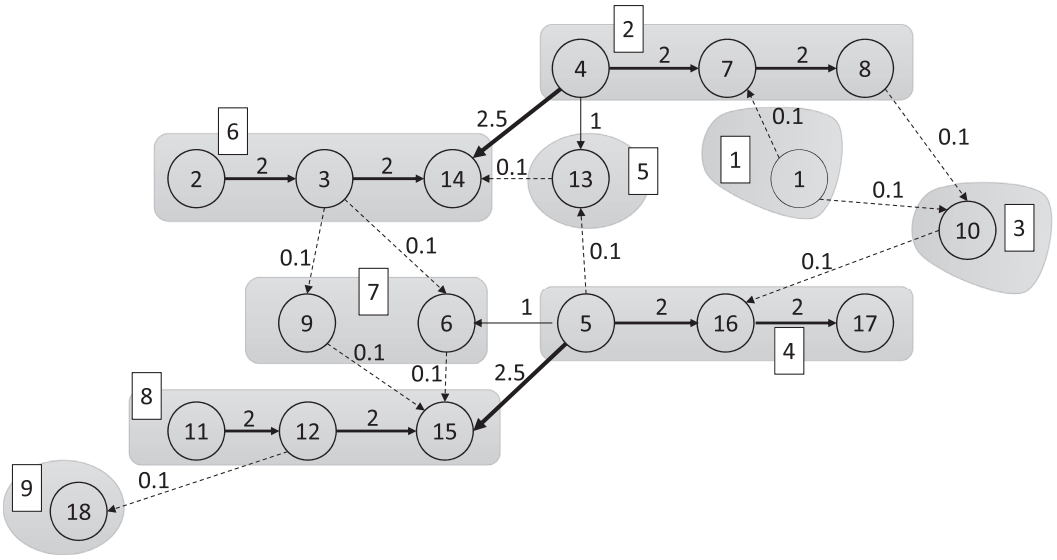

Fig. 5. Optimal solution of the instance from Fig. 3.

and Pesch (2014a). This set contains a total of 5760 instances, with integer data, and a number of trains (nodes) ranging between 6 and 40. The number of tracks, which gives the cluster capacity B, takes values {2, 4} for instances with no more than 16 trains, and values {6, 8} for the larger ones. Instances are grouped into two main classes according to the node weights distribution. In the first class, node weights ω_i are randomly taken from $\{1, \ldots, B\}$, while in the second class all weights are equal to 1 (each train uses one track).

The solver used in the study was FICO Xpress Mosel 64-bit v4.8.2 under operating system Ubuntu 16.04 LT. The computer was an Intel Xeon(R) CPU E5-2623 v3, 3.00 gigahertz with 16 gigabyte of memory. Unless stated otherwise, default values were used for all but one of the parameters of Xpress. The only modification was to discard nodes of the branching tree when the difference between their upper bound and the incumbent was at most 0.99, since the optimal values of our instances are always integer. Although the binarity of the z-variables is guaranteed at any optimal solution to formulations (P) and (P'), we included the corresponding constraints in the formulation since the solver seemed to perform better this way. The results from Section 6.1 were applied to all instances to fix to zero some x- and z-variables.

We aim to compare our results with those obtained in Nossack and Pesch (2014a) using both, formulation (NP) and a customized branch-and-bound algorithm. The results reported in that paper were obtained on a PC with an Intel Pentium Core2Duo processor at 2.2 gigahertz with 4 gigabyte of memory; the formulation was solved using CPLEX 12.4 Concert Technology and the algorithm was implemented in Java2 under Windows XP. In order to compare computational times in the fairest possible way, we have implemented formulation (NP) in our solver and solved the instances with our computer. In Nossack and Pesch (2014a), they were able to solve to optimality instances with up to 12 nodes (random weights) or 8 nodes (unit weights). With a limit time of 300 seconds, running instances with up to 32 nodes, we obtained the results presented in Table 1. Every row contains the averaged results of 240 instances. The given columns provide the number of nodes, optimal value, B, number of edges, optimal number of clusters, time in seconds reported in Nossack and Pesch (2014a), time in seconds obtained with our computer and, for the sake of completeness, number of instances that could be solved to optimality with our computer in 300 seconds. Note that the computational times with the computer used in Nossack and Pesch (2014a) range between 1 and 3 times the computational times obtained with our computer. This ratio should be taken into account in the rest of the computational study, where the times are the original ones presented in Nossack and Pesch (2014a).

Table 1
Comparing times of formulation (NP) in different computers.

n	OPT	B	m	Clusters	Times (s)		Solved
				k_{opt}	Theirs	Ours	Ours
Unit weights							
6	104.16	3.0	11.7	2.6	0.28	0.09	240
8	127.08	3.0	19.1	3.3	0.91	0.40	240
12	162.04	3.0	35.6	5.1	–	6.51	240
16	196.21	3.0	53.4	6.7	–	–	189
20	425.40	7.0	72.0	3.7	–	–	77
24	477.67	7.0	90.4	4.0	–	–	3
28	522.90	7.0	110.7	4.9	–	–	0
30	546.48	7.0	120.4	5.2	–	–	0
32	575.78	7.0	130.4	5.5	–	–	0
Random weights							
6	29.25	3.0	11.7	4.8	0.19	0.10	240
8	39.45	3.0	19.1	6.3	0.46	0.61	240
12	44.59	3.0	35.6	9.5	9.89	3.83	240
16	58.91	3.0	53.4	12.5	–	–	233
20	99.24	7.0	72.0	14.0	–	–	77
24	115.82	7.0	90.4	16.8	–	–	97
28	122.95	7.0	110.7	19.6	–	–	26
30	132.15	7.0	120.4	20.9	–	–	16
32	143.08	7.0	130.4	22.2	–	–	6

8.1. Preliminary study

In order to find a robust configuration of formulations (P) and (P') we randomly generated two new sets of instances; one with unit weights, and another one with random weights. To this end, we initially chose three parameters: n, B and a probability p. Then, we generate the weight for each node. In the first case, it is set to 1, in the second case, it is drawn from the set $\{1, \ldots, B\}$ according to the probability distribution $P(i) = K/i, i = 1, \ldots, B$, where $K = \left(\sum_{i=1}^{B} 1/i\right)^{-1}$. To generate the graph, each pair $\{i, j\}$ with $i < j$ is taken in turn, and arc (i, j) is generated with probability p. Only

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

Table 2
Previous results of variants of formulation (P).

Inequalities	zLP	BnB nodes	Time(s) all	# solved	Time(s) solved
None	268.21	83407.98	76.06	108	17.81
(32)	268.21	63137.38	74.10	109	21.00
(33)	268.21	72536.44	76.73	108	18.55
(34)	156.27	5535.77	12.60	119	7.67
(35)	217.93	16033.12	25.70	117	10.97
(37)	255.55	52505.08	60.10	112	21.52
(38)	268.21	59826.11	71.11	109	17.70
(39)	266.65	10874.59	56.89	114	28.27
(40)	265.22	9649.21	58.55	113	24.98
(41)	250.30	2755.07	19.47	120	19.47
(42)	251.85	3385.09	21.90	120	21.90
(43)	268.21	87485.77	76.39	111	33.91
(34)(35)	186.15	3057.28	10.47	120	10.47
(32)(34)(35)	178.45	2130.57	5.96	120	5.96
(33)(34)(35)	184.39	3037.88	8.95	120	8.95
(34)(35)(37)	186.11	3364.60	10.54	120	10.54
(34)(35)(43)	186.15	4579.06	9.19	120	9.19
(34)(35)(38)	185.89	1737.30	6.94	120	6.94
(32)(33)(34)(35)	177.64	1694.80	5.20	120	5.20
(32)(33)(34)(35)(37)	177.63	2439.98	6.77	120	6.77
(32)(33)(34)(35)(43)	177.64	2116.02	6.49	120	6.49
(32)(33)(34)(35)(38)	177.60	1507.19	5.07	120	5.07
(32)(33)(34)(35)(37)(43)	177.63	2311.25	6.60	120	6.60
(32)(33)(34)(35)(37)(38)(43)	177.59	1293.42	4.71	120	4.71

arcs (i, j) with $i < j$ are considered to ensure that the obtained graph is acyclic. If, after considering all pairs $\{i, j\}$ the graph is not yet connected, new arcs linking different components are added at random until the graph is connected.

For this preliminary analysis, instances with $n \in \{15, 30\}$, $B = 7$ and probabilities $p \in \{0.3, 0.5, 0.7\}$ have been generated. For each combination, we generated 20 instances, 10 with unit weights, and 10 with random weights, giving a total of 120 instances.

Table 2 shows the progressive results we obtained using different sets of valid inequalities to formulation (P) on these instances. Column zLP is the optimal value of the linear relaxation of the instance, i.e., the upper bound on the optimal value provided by (P) (the average optimal value is 156.27). The number of nodes of the branching tree and the time in seconds needed to optimally solve the instances are presented in the next two columns. Not all the instances could be solved in the maximum allowed time (fixed to 600 seconds). The number of solved instances is shown in the next column, and the last column presents the time in seconds required to solve the solved instances. When none of the families of valid inequalities were added to (P), the average time was 18 seconds for the solved instances, and 12 of them could not be solved. The number of nodes more than 83000 and the upper bound 268.21. At the end of the study all the instances could be solved, the time was reduced to 5 seconds, the number of nodes to 1300 and the upper bound was 177.6. First, we compared the effect of using each family of constraints separately. There were two families of constraints, (34) and (35), that were able to reduce the times and number of nodes, improving at the same time the bounds. Then we combined these two families with others, and continued combining families that were promising in terms of times, nodes and/or bounds. The two combinations providing the best average time are those given in the last row, and the third row from the end of the table. For these two best combinations, we also computed the median and the first and third quantile of the solution times. For the combination {(32)(33)(34)(35)(38)} these values (Q1/Me/Q3) were 0.10/0.60/4.75, while for the combination {(32)(33)(34)(35)(37)(38)(43)} they were 0.18/0.60/5.20. (The larger average value for the first of these two combinations is due to the large values of some outliers). The above values, together with the fact that the first of the two combinations includes less constraints, has led us to choose combination {(32)(33)(34)(35)(38)}.

In the case of (P'), the results we obtained with this formulation and different combinations of valid inequalities are depicted in Table 3, which has the same structure as the previous one. Constraints (50) and (51), a slight modification and an extension of constraints (34) and (35), respectively, were again selected as two of the three best options in the first stage of the study. Afterwards, by adding other families of constraints, additional improvements were reached and the best choice was decided to be the third row from the end, using constraints (48), (49), (50) and (51). Again, this combination is not the one giving the smallest average time, but the small difference is compensated by the fact that less inequalities need to be separated. Now Q1 and Q3 were 0.10 and 1.23 for both combinations, and they only differed in the median, that was 0.40 for the third row from the bottom, and 0.30 for the second one from the bottom.

The overall average computing time was 1.81 seconds, below the time required by (P) with the best combination of inequalities. The average number of nodes in the search tree was 55, much less than in the case of (P), although (P) took advantage of its reduced size. The average upper bound was 163, almost equal to the average optimal value 156.27.

8.2. Complete study

The results of applying formulations (P) and (P') to all the 2880 instances with random weights, using the best combination of valid inequalities, are shown in Table 4. Instances have been grouped according to the number of nodes and compared with the results obtained in Nossack and Pesch (2014a) using formulation (NP) as well as their ad-hoc branch-and-bound algorithm (columns NPBB). There are 12 different sizes, ranging from 6 to 40 nodes, and the averaged optimal value, capacity of the clusters and number of arcs (m) are shown in the first columns. Then, k indicates the value obtained in Proposition 6.1 and k_{opt} is the number of clusters in the optimal solution. Under "UB" we summarize the upper bounds obtained with different methods: The specific method designed in Nossack and Pesch (2014a), the linear relax-

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

Table 3
Results of variants of formulation (P') on previous instances.

Inequalities	zLP	BnB nodes	Time(s) all	# solved	Time(s) solved
None	247.90	32455.47	31.41	118	21.77
(39)	246.80	4016.95	67.00	111	23.63
(40)	246.33	5378.29	55.16	113	21.65
(41)	241.06	1177.49	34.67	116	15.08
(43)	247.90	18499.39	53.71	113	19.81
(48)	247.90	10313.57	75.77	110	27.98
(49)	247.90	19636.30	57.40	112	18.58
(50)	186.51	1841.13	11.69	119	6.75
(51)	219.60	1917.93	14.65	119	9.73
(52)	237.18	20339.11	38.55	115	14.10
(53)	247.90	26988.09	44.52	114	15.26
(50)(51)	183.62	1079.27	4.11	120	4.11
(48)(50)(51)	167.21	149.48	4.01	120	4.01
(48)(49)(50)(51)	163.39	55.36	1.81	120	1.81
(48)(49)(50)(51)(52)	163.39	55.36	1.78	120	1.78
(43)(48)(49)(50)(51)(52)	163.39	37.41	1.85	120	1.85

Table 4
Results on instances with random weights.

n	OPT	B	m	Clusters		UB			BnB nodes					Times (s)						Trimmed (s)	
				k	k_{opt}	NPBB	(P)	(P')	NPBB	(P)	(P')	(Pc)	(P'c)	(NP)	NPBB	(P)	(P')	(Pc)	(P'c)	(P)	(P')
6	29.25	3.0	11.7	5.6	4.8	29.3	29.3	29.3	1	1	1	1	1	0.19	0.01	0.00	0.00	0.00	0.00	0.00	0.00
8	39.45	3.0	19.1	7.5	6.3	39.6	39.6	39.6	1	1	1	1	1	0.46	0.02	0.00	0.00	0.00	0.00	0.00	0.00
12	44.59	3.0	35.6	11.4	9.5	45.1	44.8	44.7	1	1	1	1	1	9.89	0.09	0.00	0.00	0.00	0.00	0.00	0.00
16	58.91	3.0	53.4	15.3	12.5	59.6	59.4	59.3	2	1	1	1	1	–	0.37	0.01	0.01	0.01	0.03	0.01	0.01
20	99.24	7.0	72.0	19.1	14.0	101.9	101.8	101.0	4	3	2	4	2	–	1.40	0.22	0.21	0.20	0.18	0.20	0.21
24	115.82	7.0	90.4	22.9	16.8	119.7	119.6	118.4	7	29	4	24	3	–	2.79	0.54	0.55	0.52	0.45	0.52	0.54
28	122.95	7.0	110.7	26.9	19.6	127.2	127.6	126.2	11	70	29	84	6	–	4.93	1.50	1.42	1.38	1.10	1.47	1.38
30	132.15	7.0	120.4	28.8	20.9	137.4	138.0	136.1	17	640	154	271	9	–	6.84	2.64	2.27	1.99	1.52	2.44	2.18
32	143.08	7.0	130.4	30.7	22.2	149.0	149.5	147.6	21	374	106	289	25	–	9.10	3.50	3.21	2.66	2.13	3.28	3.15
34	151.23	7.0	140.8	32.9	23.6	157.3	158.4	156.1	20	772	228	601	29	–	11.14	5.63	4.78	4.00	2.79	4.97	4.58
36	157.08	7.0	149.7	34.7	24.8	163.5	164.3	162.0	25	1135	234	874	25	–	14.83	7.74	6.13	5.13	3.61	7.19	5.94
40	165.93	7.0	170.5	38.7	27.8	173.7	175.1	172.5	47	3733	1174	1942	200	–	29.20	22.46	14.47	12.22	7.34	18.50	13.46

ation of formulation (P) and the linear relaxation of formulation (P'). Under "BnB nodes" we specify the number of nodes of the different branch and bound algorithms. Computational times are compared in the following six columns, under "Times (s)". Column (NP) refers to the best formulation known so far, that could be used in Nossack and Pesch (2014a) only to solve instances with up to 12 nodes. Note that both formulations developed in this article, (P) and (P'), solve all the instances with low computational effort with only one exception: Among the 2880 instances, (P) required more than fifteen minutes to solve one of them (of size $n = 40$). The maximum time in the case of (P') was below four minutes. Formulation (P') is larger than formulation (P), but since it produces better upper bounds, the number of nodes in the search tree is very small and this gives rise to the best computational results. Even when compared with the ad-hoc algorithm (column NPBB under "Times (s)") the results are very good. Recall that the NPBB and NP columns have been directly taken from Nossack and Pesch (2014a). Since there were a few instances that we could classify as *outliers*, the last columns of Table 4 ("Trimmed (s)") show the averaged times, removing the instance with largest computational time for each value of n.

Finally, we considered the separation of valid inequalities (36) in formulation (P), columns (Pc) of Table 4. We still used inequalities (32), (33), (34) and (38) but removed (35), since they are part of family (36). We also switched off the separation of the solver own cuts. The average number of cuts per instance was small. In the case of $n = 40$ this mean was 8. But this small number of cuts produced an average reduction of the number of nodes in the branching tree in all rows of the table. We observed that the computational times improved considerably with the new approach. A deeper analysis led us to conclude that the cuts added by the solver were slowing down the process. Although constraints (36) have no effect when constraints (51) are combined with the rest of valid inequalities in formulation (P'), and thus their separation has no sense in this case, we gave (P') the opportunity of run without the solver's cuts, and named (P'c) the columns with the corresponding results. Note that, in general, the number of nodes of the search tree and the computational times dipped to a record low.

The results with the best combination of inequalities over the 2880 instances with unit weights are shown in Table 5.

In this case (NP) could be used in Nossack and Pesch (2014a) only to solve instances with up to 8 nodes and the ad-hoc branch-and-bound algorithm could not solve instances with 32 to 40 nodes. In general, these instances were more difficult to solve, and all formulations produced worse upper bounds and needed larger computational times.

When we included the separation of valid inequalities (36) in formulation (P), the average number of cuts per instance in the largest instances with $n = 40$ was 92, much larger than in the case of random weights. Since we prevented the solver from using its own cuts, we incorporated again the most promising valid inequalities. The effect was a significant reduction of the computational times.

Regarding (P'), the bounds were again quite better but the computational times were affected by several instances that required many branching nodes and large computational times (in particular, four of them took more than 1000 seconds). But the removal

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022

Table 5
Results using unit weights.

n	OPT	B	m	Clusters		UB			BnB nodes					Times (s)					
				k	k_{opt}	NPBB	(P)	(P')	NPBB	(P)	(P')	(Pc)	(P'c)	(NP)	NPBB	(P)	(P')	(Pc)	(P'c)
6	104.16	3.0	11.7	3.5	2.6	104.8	105.1	104.4	1	1	1	0.72	0.93	0.28	0.01	0.00	0.00	0.00	0.00
8	127.08	3.0	19.1	4.0	3.3	127.5	128.2	127.2	1	1	1	0.93	0.96	0.91	0.02	0.00	0.00	0.00	0.00
12	162.04	3.0	35.6	6.5	5.1	164.1	167.6	163.4	3	2	1	2.87	1.35		0.12	0.04	0.03	0.02	0.01
16	196.21	3.0	53.4	9.0	6.7	198.9	205.1	198.1	5	10	1	14.19	1.83		0.52	0.25	0.17	0.09	0.07
20	425.40	7.0	72.0	5.0	3.7	438.5	469.0	433.4	15	23	4	31.35	5.30		2.56	0.72	1.42	0.27	0.67
24	477.67	7.0	90.4	6.0	4.0	492.8	540.4	488.2	33	130	9	171.73	16.03		6.45	1.68	3.65	0.59	1.54
28	522.90	7.0	110.7	7.5	4.9	546.3	606.5	540.3	88	806	40	1022.75	51.27		21.82	4.79	11.20	1.80	4.58
30	546.48	7.0	120.4	8.0	5.2	570.2	635.4	564.2	136	1378	53	1868.83	79.88		41.37	9.78	14.74	3.17	7.58
32	575.78	7.0	130.4	8.0	5.5	–	673.4	595.8	–	1673	91	2270.63	129.85		–	10.31	20.62	3.87	9.05
34	590.46	7.0	140.8	8.0	5.9	–	693.9	613.0	–	1858	87	2427.56	150.85		–	11.54	23.26	4.69	8.94
36	617.88	7.0	149.7	9.5	6.3	–	737.1	644.4	–	9090	259	12384.84	422.60		–	60.98	44.81	22.59	17.85
40	663.42	7.0	170.5	10.0	6.9	–	806.6	695.6	–	13985	262	18802.05	511.37		–	78.53	74.04	32.81	27.47

of the automatic cuts of the solver (column (P'c)) produced an impressive decrease of the computational times and again this option dominated the others in terms of bounds, nodes and computational times. This can be observed both, in the case of random weights and of unit weights.

All in all, the combination of the new formulations, valid inequalities, preprocessing and cuts made it possible to efficiently solve to optimality all instances generated in Nossack and Pesch (2014a), even those that were not solved before.

Regarding the comparison of our results with NPBB, we could observe in Table 1 that the cpu times required by our solver in our computer are at most one third of those obtained in Nossack and Pesch (2014a). Now it can be observed in Tables 4 and 5 that our times, for the instances that could be solved in Nossack and Pesch (2014a), are about one order of magnitude smaller in the case of unit weights, and more than one third smaller in the instances with random weights.

During the experiments we observed that the addition of some of the valid inequalities that were not selected in the previous study also contributed to the reduction of bounds and times in the case of the largest instances and they should not be left outside of the resolution of real-life instances.

9. Conclusions

In this paper we have addressed the particular case of the train scheduling problem arising at rail-rail transshipment yards when trains are allowed to enter the loading area only once. As in previous works, we modeled this problem as an acyclic graph partitioning problem, for which we have presented two variants of a new MIP formulation, based on a characterization of acyclic graphs. Additionally, several properties of the problem and its solutions are studied, which allow to derive several valid inequalities and variable fixing rules.

A thorough computational experience has been carried out on a large set of instances taken from the literature that allowed first to determine the best configuration of the formulation variants, and later to show the capability of the resulting enhanced formulations of solving instances much faster than in the previous approaches, including a tailor-made branch and bound algorithm. Indeed, we were able to solve to optimality instances that could not be solved so far.

Our computational results also confirm what was already observed in previous works concerning the difficulty of the instances; those with unit weights result much harder to solve than those with nodes of different weights.

Acknowledgments

The authors would like to thank Jenny Nossack and Erwin Pesch for kindly providing us with the data used in the computational study.

The research of Maria Albareda is partially funded by the Spanish *Ministerio de Economía y Competitividad* through project MTM2015-63779-R. Alfredo Marín acknowledges that research reported here was supported by Spanish *Ministerio de Economía y Competitividad*, project MTM2015-65915-R, *Fundación Séneca de la Consejería de Educación de la Comunidad Autónoma de la Región de Murcia*, project 19320/PI/14, and *Fundación BBVA*, project "Cost-sensitive classification. A mathematical optimization approach" (COSECLA). A.M. Rodríguez-Chía acknowledges that research reported here was supported by MTM2013-46962-C2-2-P, MTM2016-74983-C2-2-R, supported by Agencia Estatal de Investigación (AEI) and the European Regional Development's funds (FEDER).

References

Alpert, C. J., & Kahng, A. B. (1995). Recent directions in net list partitioning: a survey. *Integration, the VLSI Journal, 19*, 1–81.
Bartolini, S., Casini, I., & Detti, P. (2014). Solving graph partitioning problems arising in tagless cache management. *Lecture Notes in Computer Science, 8596*, 50–61.
Benati, S., Puerto, J., & Rodríguez-Chía, A. M. (2017). Clustering data that are graph connected. *European Journal of Operational Research, 261*, 43–53.
Boysen, N., Fliedner, M., Jaehn, F., & Pesch, E. (2013). A survey on container processing in railway yards. *Transportation Science, 47*, 312–329.
Boysen, N., Jaehn, F., & Pesch, E. (2011). Scheduling freight trains in rail-rail transshipment yards. *Transportation Science, 45*, 199–211.
Cong, J., Li, Z., & Bagrodia, R. (1994). Acyclic multi-way partitioning of boolean networks. In *Proceedings of the 31st annual design automation conference* (pp. 670–675). New York: ACM.
Dorndorf, U., & Pesch, E. (1994). Fast clustering algorithms. *ORSA Journal on Computing, 6*, 141–153.
European Commission. (2011). *White paper on transport—roadmap to a single European transport area—towards a competitive and resource-efficient transport system*. Luxembourg: Publications Office of the European Union. doi:10.2832/30955.
Eurostat Statistics Explained. (2017) http://ec.europa.eu/eurostat/statistics-explained/index.php/Freight_transport_statistics.
Ferreira, C. E., Martin, A., de Souza, C. C., Weismantel, R., & Wolsey, L. A. (1996). Formulations and valid inequalities for the node capacitated graph partitioning problem. *Mathematical Programming, 74*, 247–266.
Ferreira, C. E., Martin, A., de Souza, C. C., Weismantel, R., & Wolsey, L. A. (1998). The node capacitated graph partitioning problem: a computational study. *Mathematical Programming, 81*, 229–256.
Fiduccia, C. M., & Mattheyses, R. M. (1982). A linear-time heuristic for improving network partitions. In *Proceedings of the 19th annual design automation conference* (pp. 175–181). New York: IEEE Press.
Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability – a guide to the theory of NP-completeness*. New York: Freeman.
Grötschel, M., & Wakabayashi, Y. (1989). A cutting plane algorithm for a clustering problem. *Mathematical Programming, 45*, 59–96.
Grötschel, M., & Wakabayashi, Y. (1990). Facets of the clique partitioning polytope. *Mathematical Programming, 47*, 367–387.
Holm, S., & Sørensen, M. M. (1993). The optimal graph partitioning problem. *OR Spectrum, 15*, 1–8.

Jaehn, F., & Pesch, E. (2013). New bounds and constraint propagation techniques for the clique partitioning problem. *Discrete Applied Mathematics, 161*, 2025–2037.
Ji, X., & Mitchell, J. E. (2007). Branch-and-price-and-cut on the clique partitioning problem with minimum clique size requirement. *Discrete Optimization, 4*, 87–102.
Johnson, E. L., Mehrotra, A., & Nemhauser, G. L. (1993). Min-cut clustering. *Mathematical Programming, 62*, 133–151.
Kernighan, B. W., & Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal, 49*, 291–307.
Lukes, J. A. (1974). Efficient algorithm for the partitioning of trees. *IBM Journal of Research and Development, 18*, 217–224.
Mehrotra, A., & Trick, M. A. (1998). Cliques and clustering: a combinatorial approach. *Operations Research Letters, 22*, 1–12.
Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM, 7*, 326–329.
Miyauchi, A., & Sukegawa, N. (2015). Redundant constraints in the standard formulation for the clique partitioning problem. *Optimization Letters, 9*, 199–207.
Nobibon, F. T., Hurkens, C., Leus, R., & Spieksma, F. C. R. (2010). Exact algorithms for coloring graphs while avoiding monochromatic cycles. *Lecture Notes in Computer Science, 6124*, 229–242.
Nobibon, F. T., Hurkens, C., Leus, R., & Spieksma, F. C. R. (2012). Coloring graphs using two colors while avoiding monochromatic cycles. *INFORMS Journal on Computing, 24*, 485–499.
Nossack, J., & Pesch, E. (2014a). A branch-and-bound algorithm for the acyclic partitioning problem. *Computers & Operations Research, 41*, 174–184.
Nossack, J., & Pesch, E. (2014b). Mathematical formulations for the acyclic partitioning problem. In *Proceedings of the operations research proceedings* (pp. 333–339). Springer.
Oosten, M., Rutten, J. H. G. C., & Spieksma, F. C. R. (2001). The clique partitioning problem: facets and patching facets. *Networks, 38*, 209–226.
Parrochia, D., & Neuville, P. (2013). *Towards a general theory of classifications*. In Birkhäuser (Ed.).
Sanchis, L. A. (1989). Multiple-way network partitioning. *IEEE Transactions on Computers, 38*, 62–81.
Wong, E. S. H., Young, E. F. Y., & Mak, W. K. (2003). Clustering based acyclic multiway partitioning. In *Proceedings of the 13th ACM great lakes symposium on VLSI* (pp. 203–206). ACM.

Please cite this article as: M. Albareda-Sambola, A. Marín and A.M. Rodríguez-Chía, Reformulated acyclic partitioning for rail-rail containers transshipment, European Journal of Operational Research, https://doi.org/10.1016/j.ejor.2019.02.022