Scheduling Container Movements per Crane in Train-Train Transshipment Terminals Using Simulated Annealing

Sam Heshmati¹, Maria Antónia Carravilla², José Fernando Oliveira³

INESC TEC, Faculdade de, Engenharia, Universidade do Porto, Portugal

prdeig1103803@fe.up.pt, mac@fe.up.pt², jfo@fe.up.pt³

Abstract: - Train-train transshipment terminals are used to transship containers among trains. Scheduling container movements per crane (SCMC) is one of the sub-problems in train-train transshipment. The objective is to determine the sequence of container movements for each crane such that all containers are positioned on the appropriate train or on the yard, while minimizing the makespan. This study analyzes the sequence of container transshipment per crane in modern train-train transshipment terminals. We propose a simulated annealing (SA) based heuristic for solving the SCMC. The proposed SASCNC heuristic is tested on four sets of instances and the results are presented. The computational results show that the proposed algorithm improves the solution more than 15%.

Keywords: Railway systems, Transshipment yards, Container handling, Crane scheduling, simulated annealing.

1. Introduction

Freight transport is a critical factor in economic development and it is executed in different modes such as air network, sea network, or inland freight networks. Inland freights are usually transported either by rail or
by trucks. It is desirable to increase rail transport for several reasons, namely the increase in transport per rail for medium or long distances provides the opportunity to relieve congested roads and can impact the environmental issues through the reduction in CO$_2$ emissions [1]. To achieve a higher share of freight traffic on rail, there is a need for a higher efficiency in freight handling in train terminals. Therefore, it requires technical innovation as well as better decision making through the development of suited decision support systems.

This study analyzes the sequence of container transshipment per crane in modern train-train transshipment terminals. Modern train-train transshipment terminals consist of a number of rail tracks, where freight trains come consecutively in bundles to be served (one train per track). Traditional shunting yards, where wagons are exchanged via shunting hills and switches among trains, considerably slow down the transshipment process and, therefore, may be the cause for late deliveries. In modern terminals, rail-mounted gantry cranes move containers between different freight trains, without exchanging wagons [2]. Figure 1 gives a schematic representation of such a train-train transshipment terminal.

![Figure 1: A schematic View of Train-Train Transshipment Terminal](image-url)
There are several decision problems and challenges in this type of transshipment terminals. Studies in literature decomposed the train-train transshipment problem into different sub-problems. The sub-problems consist in assigning destinations to trains, scheduling the service slots, assigning parking position for trains, determining the destination positions of containers, assigning container moves to cranes, and determining the sequence of container moves per crane.

In this study we are aiming at developing a meta-heuristic based on simulated annealing to solve the problem of determining the sequence of container movements per crane (SCMC) in train-train transshipment. The objective is to determine the sequence of container movements for each crane such that all containers are positioned on the appropriate train or on the yard, while minimizing the make-span and the empty movements of the crane.

This paper is structured as follows. In the next section the related literature is shortly reviewed. Then, in Section 3 we overview the problem and its characteristics. Afterwards, the proposed heuristic is explained in Section 4. Finally, the results are presented and discussed; later on the conclusion and opportunities for future work are explained.

2. Related Literature
The studies related to scheduling the crane movements in container terminals have been mainly carried out by using mathematical programming techniques (e.g. integer programs or mixed integer programs) or by proposing heuristics and metaheuristics to solve real world large scale problems. Alicke [3] studies the problem of scheduling the sequence of the container transshipment in Mega Hub in Germany by using constraint satisfaction programming. A given set of crane moves is assigned to cranes with overlapping areas of operation, which are blocked whenever a crane enters an area. In another study, Souffriau et al. [4] decomposed the train-train transshipment problem into different sub-problems. They used
Variable Neighborhood Descent metaheuristic to sequence the container moves per crane.

3. Problem Description

This section describes the different entities of the problem: A terminal in this problem consists of a number of parallel tracks, a buffer lane, gantry cranes and a yard. The tracks and buffer lane are characterized by a Y coordinate. Trains are in the terminal and the destination of each train has been assigned. A train is composed of a number of wagons which hold containers. All containers are of the same type and the X coordinate of the first one is 1, while the X coordinate of $i^{th}$ container is $i$. The problem assumes the cranes move in lateral and longitudinal direction, simultaneously and with the same speed. Therefore the Chebyshev distance (i.e. the maximum of the lateral and longitudinal distance) between the destination of the first container and the origin of the next container corresponds to the travelling time between two transshipments. This problem considers a static crane assignment which means that each crane has a disjoint area of operations assigned, and all container moves falling into that area are exclusively processed by the respective crane [5]. In this problem the positions of the containers on its inbound train, as well as the position of the containers on the out-bound train or yard are given. Since the origin position and the destination position of the containers are determined, the travelling time of a crane is fixed while carrying a container, therefore it is enough to minimize the time of empty travelling of the crane in order to minimize the total transshipment time. In other words the strategy for minimizing the makespan is to sequence the transshipment jobs such that the empty travelling time of the crane, from one transshipment to the other is minimized.

A transshipment job can be carried out only when the destination position is free. Based on the status of the destination of the container, three types of movements may be identified. Type I is a simple movement where the destination position is empty, and therefore no
precedence needs to be considered. Type II can only be executed after a previous movement, as the destination is not empty at the beginning. This happens when the destination of one transshipment, overlaps with the origin of the other one. In this case the crane should first carry out the transshipment that results in the evacuation of the occupied position, and afterwards the locked transshipment is executable. Type III is called deadlock transshipment. Deadlock happens when the origin of one transshipment is the destination of the other one and vice versa, in other words when the crane needs to swap the position of two containers. Figure 2 illustrates the three types of transshipment.

To solve a deadlock the algorithm needs to split the two transshipments into three transshipments. To start the crane needs to put the first container into the buffer lane, next the second container would be transshipped into its destination, and finally the crane transships the first container from the buffer lane to its destination. As a result, for solving the deadlock situation the algorithm splits it into three transships with precedence.

To determine which container should go to the buffer lane, we need to compute the empty and loaded travelling time of the crane in both cases and choose the container which has the minimum
time span. Assume there are two containers in $D_1 = (x_1, y_1)$, $D_2 = (x_2, y_2)$ and $y = y_b$ is the buffer lane. The crane has to swap the containers, for this purpose it needs to move one of the containers to the buffer lane in order to free the position and move the second container to the free place and afterwards move the first container from buffer to its destination position.

**Case 1: if $D_1$ goes to the buffer lane**

The transshipments with precedence:

$$
\begin{align*}
D_1, & \text{ from } (x_1, y_1) \text{ to } (x_1, y_b) \\
D_2, & \text{ from } (x_2, y_2) \text{ to } (x_1, y_1) \\
D_1, & \text{ from } (x_1, y_b) \text{ to } (x_2, y_2)
\end{align*}
$$

(1)

**Case 2: if $D_2$ goes to the buffer lane**

The transshipments with precedence:

$$
\begin{align*}
D_2, & \text{ from } (x_2, y_2) \text{ to } (x_2, y_b) \\
D_1, & \text{ from } (x_1, y_1) \text{ to } (x_2, y_2) \\
D_2, & \text{ from } (x_2, y_b) \text{ to } (x_1, y_1)
\end{align*}
$$

(2)

Based on the value of $|x_1 - x_2|$, $(y_b - y_1)$, and $(y_b - y_2)$ There are three states to make a decision on which container goes to the buffer lane and this movement imposes less travelling (i.e. empty and loaded) to the crane:

**First: if**

$$
\min(|x_1 - x_2|, (y_b - y_1), (y_b - y_2)) = (y_b - y_2)
$$

Travelling time of crane is greater in case 1, so the algorithm chooses $D_2$ to go to the buffer lane.

**Second: if**

$$
\min(|x_1 - x_2|, (y_b - y_1), (y_b - y_2)) = (y_b - y_1)
$$

Travelling time of crane is greater in case 2, so the algorithm chooses $D_1$ to go to the buffer lane.

**Third: if**

$$
\min(|x_1 - x_2|, (y_b - y_1), (y_b - y_2)) = |x_1 - x_2|
$$

The empty and loaded travelling time of the crane is the same in both cases. Therefore there is no preference and it doesn't matter which container goes to the buffer lane.

After this pre-processing phase we have a list of transshipments and a list of precedencies to be met to guarantee the admissibility of the solutions.
4. Proposed Heuristic Algorithm

We propose SASCMC heuristic for the SCMC, based on simulated annealing. Simulated annealing is a metaheuristic that is capable of escaping from being trapped into a local optimum by accepting worse solutions during its iterations. Next, we describe the proposed SASCMC heuristic in detail, including the constructive heuristic to generate an initial solution, the neighborhood, the evaluation function, the parameters used, and the SASCMC procedure.

Constructive heuristic: The problem is modeled as a Sequential Ordering Problem (SOP), which is a version of the Asymmetric Traveling Salesman Problem (ATSP) where precedence constraints must also be observed [6]. In this problem, the crane starts from a random container, does the transshipment jobs in sequence and finally returns to the first point. The construction of the initial solution is carried out by using a greedy algorithm based on the travelling time between the point where the crane is and the position of each container. The crane does the transshipment of the nearest container. There is also a “precedence list” which reflects the locks and deadlocks as precedence constraints. If the selected transshipment has a precedence to consider, the algorithm checks if the precedence constraints are satisfied or not. If the precedence constraints are satisfied the transshipment is chosen, but if not, the next transshipment that satisfies the precedence constraints will be chosen. A solution which includes all the transshipments with no precedence constraint violation is a feasible solution.

Neighborhood: We defined 2-Swap Transshipment neighborhood. Every feasible solution generated by swapping two transshipments of the current solution belongs to its neighborhood. To check the feasibility of the generated solution after swapping two transshipments α and γ, the algorithm checks if transshipments α or γ are in the “precedence list”. If one or both of the transshipments α and γ
are in the “precedence list”, the algorithm checks all the transshipments between these two transshipments. If there is any transshipment job which by swapping transshipments α and transshipment γ, the precedence constraint is violated and the solution is infeasible. The solutions with violated precedence constraints will be removed. In this case the size of the neighborhood is equal to \( n(n-1)/2 \) where \( n \) represents the number of transshipment jobs (i.e. movement of containers).

**Evaluation function:** The algorithm uses the following evaluation function in order to compare the solutions in the neighborhood with the current solution, where: \( v \) is the current solution and \( x_{ij} \) is 1 if transshipment \( i \) is followed by transshipment \( j \) and 0 otherwise.

\[
c_{ij} = \text{time of making transshipment } i \text{ followed by transshipment } j
\]
\[
eval(v) = \sum x_{ij} c_{ij}
\]

(3)

**Parameter settings:** The SASCMC uses four parameters \( I_t, T_0, T_f, \) and \( \alpha \). \( I_t \) represents the number of iterations at a particular temperature. \( T_0 \) denotes the initial temperature, while \( T_f \) is the freezing temperature below which the SASCMC procedure is terminated. Finally, \( \alpha \) is the cooling ration.

**The SASCMC procedure:** Firstly, the temperature \( T \) is initialized to \( T_0 \). Then an initial solution \( v_c \) is generated using the greedy procedure described above. At each iteration, the SASCMC generates random feasible solution \( v_n \) as many as 20% of the size of the neighborhood and next candidate \( v_{n-best} \) is chosen by evaluating among them which returns best \( eval(v_n) \)

\[
\Delta = \eval(v_{n-best}) - \eval(v_c)
\]

Let \( \Delta = \eval(v_{n-best}) - \eval(v_c) \). The probability of replacing \( v_c \) with \( v_{n-best} \), given that \( \Delta > 0 \), is \( e^{-\Delta/T} \). It is accomplished by generating a random number \( r \in [0,1] \) and replacing the solution \( v_c \) with \( v_{n-best} \) if \( < e^{-\Delta/T} \). Meanwhile, if \( \Delta \leq 0 \), \( v_c \) is replaced with \( v_{n-best} \). The best solution found so far is recorded in
\( v_{best} \) as the algorithm progresses. The current temperature \( T \) is decreased after running \( I_t \) iterations, and replaced by \( \alpha T \). The algorithm is terminated when the current temperature \( T \) is lower than \( T_f \).

5. Results

The proposed SASCMC was implemented in C++ language and run on a PC with an Intel Core i5-2400 CPU (3.1 GHz) and 8 GB memory. A set of four problem instances is used to verify the proposed SASCMC algorithm. Since there was no instance with the characteristics that we need for our problem in the literature, the instances are generated by a random generator. In our approach, the parameters of the SASCMC are chosen by experimentation and different combinations of parameters were tested. Finally, and \( \alpha=0.98 \) were chosen due to better performance. The current temperature will fall below after 148 iterations and the algorithm will be terminated.

Table 1 illustrates the results obtained. Let number of transshipments in an instance.

<table>
<thead>
<tr>
<th>Instance</th>
<th>No. of transshipment</th>
<th>No. of deadlock</th>
<th>Greedy Average</th>
<th>SASCMC Average</th>
<th>Improvement</th>
<th>Run time (average)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance 01</td>
<td>33</td>
<td>0</td>
<td>72.522</td>
<td>54.348</td>
<td>25.06%</td>
<td>28.12(s)</td>
</tr>
<tr>
<td>Instance 02</td>
<td>33</td>
<td>4</td>
<td>71.44</td>
<td>58.44</td>
<td>18.20%</td>
<td>32.87(s)</td>
</tr>
<tr>
<td>Instance 03</td>
<td>45</td>
<td>3</td>
<td>97.172</td>
<td>82.862</td>
<td>14.73%</td>
<td>110.49(s)</td>
</tr>
<tr>
<td>Instance 04</td>
<td>60</td>
<td>5</td>
<td>196.235</td>
<td>139.441</td>
<td>28.94%</td>
<td>469.67(s)</td>
</tr>
</tbody>
</table>

Table 1: sequencing container movements results

Each instance has been run \( \mu \) times, in order to have all the transshipments assigned as a starting point once. The average, best, and worst solution is shown in the table. By increasing the size of the instances the run time increases. As it is shown in the table, SASCMC improved the solution between 14 -29 percent compared to the greedy algorithm. We can conclude that if the
number of deadlocks, as a percentage of the number of transshipments increases there are less admissible solutions and the improvements of the meta-heuristic are less.

6. Conclusion

In this paper, we proposed an SASCMC heuristic for the SCMC, based on the simulated annealing heuristic in train-train transshipment terminals. The problem was formulated as sequential ordering problem and the SASCMC algorithm performs well in terms of solution quality. Its achievement was remarkable in finding better solutions to instances with large size. On the other hand time consuming is one of the drawbacks of simulated annealing algorithms. In the real world problem normally there is more than one crane with overlapping the working area in transshipment yards. Further research will mainly focus on solving this problem with more than one crane.

7. Acknowledgement

This work is partially financed by the ERDF – European Regional Development Fund through the COMPETE Program (operational program for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project «FCOMP-01-0124-FEDER-037281».

References


Authors profile:

Sam Heshmati is PhD student at the Department of Industrial Engineering and Management of the Faculty of Engineering of the University of Porto and Researcher in the Industrial Engineering and Management Unit at INESC TEC, in Portugal. His current research interests are Combinatorial Optimization, Simulation Optimization, and Container handling logistics.

Maria Antónia Carravilla is Associate Professor at the Faculty of Engineering of the University of Porto, at the Department of Industrial Engineering and Management, and a Senior Researcher at INESC TEC. She has worked on Decision Support Systems in Operations Management and Logistics, both with metaheuristics algorithms and mathematical programming models, and has a strong track record of running R&D projects with companies and public organizations.

José Fernando Oliveira is Associate Professor at the Department of Industrial Engineering and
Management of the Faculty of Engineering of the University of Porto and Senior Researcher in the Industrial Engineering and Management Unit at INESC TEC, in Portugal. He has worked on the use of the quantitative methods, provided by Operations Research and Management Science, to support decision making in Operations Management and Logistics problems, both in industry and services.