

Peter Bohrer

Crane Scheduling in Container Terminals

Diploma thesis



Advisor

Prof. Dr. Horst W. Hamacher

August 2005

Table of Contents

1. Introduction	3
2. Related work.....	5
3. Mathematical models	6
3.1 General parameters and assumptions	6
3.2 The job models	7
3.2.1 The job model for RTGC scheduling	7
3.2.2 The job model for RMGC scheduling	11
3.3 The workload models	13
3.3.1 The workload model for RTGC scheduling	13
3.3.2 The workload model for RMGC scheduling	14
4. Complexity	15
5. Upper bound for the makespan	18
5.1 Determine an upper bound π for the RTJP	18
5.2 Determine an upper bound π for the RMJP	19
6. Solution procedures.....	21
6.1 Solution procedures for the RTJP	21
6.1.1 The Earliest Release Date - Heuristic	21
6.1.2 The Quotient-Heuristic.....	23
6.2 Solution procedures for the RMJP	24
6.3 Example.....	24
6.4 Idea for a Branch and Bound procedure.....	26
6.4.1 Lagrangian relaxation.....	26
6.4.2 The subproblem $L_1(u)$	28
6.4.3 The subproblem $L_2(u)$	29
6.5 Solution procedures for the RTWP	34
6.6 Solution procedures for the RMWP	34
6.6.1 The Lagrangian Decomposition Method for the RMWP	34
6.6.2 The Successive Piecewise-Linear Approximation Method for the RMWP.....	38
7. Lower bound.....	41
8. Approximation bounds	43
8.1 Approximation bound for the RTJP	44
8.2 Approximation bound for the RMJP	46
9. Computational experiments	48
9.1 Problem generation	48
9.1.1 Problem instances for the RTJP	48
9.1.2 Problem instances for the RMJP	49
9.2 Results	49

10. Special Cases.....	53
10.1 $K = 1$	53
10.2 $p_j = 1$	54
11. Conclusion.....	56
References	57
Picture Credits.....	59

1. Introduction

The globalization of trade has led to an enormous growth in sea transportation over the last years. A great majority of general cargo is containerized and, therefore, handled in container terminals, which are essential hubs in sea transportation systems. Besides this growth, the rising competition among the ports puts pressure on them to improve their performance.

A container terminal works under several objectives. Two of them are to minimize the turnaround time, i.e. the average period of time a vessel stays in a terminal, and to maximize the terminal throughput. The potential of cost saving is large, because of the fact that an average cargo liner spends about 60% of its time in a port (e.g. see [5]). Therefore, the terminals try to improve their turnaround time. The turnaround time and the throughput are the results of several interrelated container flows in the terminal.

A container terminal can be divided into three areas: the quayside, the container yard, and the gatehouse. At the quayside, the vessels are loaded or unloaded by quay cranes (QC). Internal trucks are used to organize the transport of the containers between the quayside and the container yard. In the yard, the containers are stored temporarily until they have to be loaded onto another vessel or until they are taken out of the terminal on the road by external trucks. Such external trucks enter the terminal through the gatehouse, which is the land entrance of the terminal. Figure 1 shows the "K" LINE Tokyo Container Terminal. This picture allows an easy identification of the three areas of a container terminal described above.



Figure1: "K" LINE Tokyo Container Terminal

1. Introduction

In the yard, containers are grouped into blocks, which consist of about 20 containers in length, six to eight containers in width and four to six containers in height. A row of such blocks is called a “yard zone”. Yard cranes are used in order to put the containers onto internal and external trucks or to unload the trucks. There are two types of yard cranes: rail-mounted gantry cranes (RMGC) and rubber-tired gantry cranes (RTGC). The RMGC have to use the rails and, therefore, can move only along an axis, i.e. within a yard zone, while the RTGC can move freely among all blocks within the whole yard. Figure 2 and 3 show how these cranes look like in reality; Figure 4 and 5 illustrate the difference in the possible movements for each crane type.



Figure 2: RTGC in the Port of Baltimore, USA



Figure 3: RMGC in the Krakow Krzeslawice Container Terminal, Poland

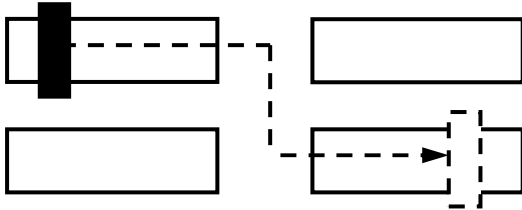


Figure 4: Possible RTGC movement

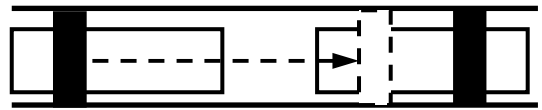


Figure 5: Possible RMGC movement

The space in a container yard is very limited. Nonetheless, thousands of containers have to be handled every day and the turnaround times should be as low as possible. Therefore, the workflow of a container terminal has to be very efficient. This is the reason why logistical planning tries to assign and to coordinate the operations of port equipments, such as berth space, quay cranes, yard space, yard cranes and trucks, as good as possible. Space allocation and truck scheduling are important tools to optimize the workflow in a container terminal, but those problems are out of the scope of this thesis. Moreover, the focus lies on the scheduling of cranes in container terminals in order to improve the turnaround time of a port. These cranes are very expensive and, therefore, their planning has strong impact on the performance of the container terminal system.

In this thesis, models to schedule two crane types (RMGC and RTGC) are developed with respect to their characteristic abilities. The suggested procedures and algorithms lead to crane schedules which make the container workflow more efficient and improve the performance of ports in a global and highly competitive economy.

2. Related work

The enormous cost saving potential makes logistic problems in a container terminal an interesting research topic. Each of the resources mentioned above (berth space, quay cranes, yard space, yard cranes and trucks) plays an important role in the processes of a container terminal. A comprehensive review of literature on several models and aspects on how to deploy these resources is published in [17].

In the following a short review of existing studies about more specific research related to terminal resources is given:

Berths are an important resource in a terminal because under heavy traffic of vessels, it is critical to allocate the berths to the vessels in order to minimize the time the vessel has to stay in the port. Research on the berth allocation problem is done in [11] and [12]. In [4], non-linear integer programming is used to analyze the problem, Chen and Hsieh work with modified time-space networks in [1], while in [10] simulation is used.

The scheduling of the container transfer between the quayside and the yard is investigated in [8].

The research on crane scheduling can also be grouped in four categories: Scheduling of RMGCs or RTGCs within a block, scheduling of RMGCs along an axis within a yard zone, scheduling of RTGCs within the whole container yard and the scheduling of QCs along the berth. The first category is analyzed in [6] and [7]. The problem is formulated for a single crane and solution methods are developed. In [9], it is tried to tackle this problem by using simulation. For the second category, Ng and Mak developed in [14] a branch and bound algorithm for a single crane problem. Ng proposed in [13] a scheduling heuristic for the multiple crane case with respect to inter-crane interference. A Lagrangean heuristic for problems of the third category is developed in [18] and [1] extends this work by another solution procedure using nonlinear programming. The problem of scheduling quay cranes along the berth is investigated in [3] and [15], where some scheduling principles are proposed in order to get good solutions for this problem.

3. Mathematical models

In this section, two different types of crane scheduling problems are modeled as integer programs. The first problem considered is the scheduling of RTGCs which can move freely among the blocks in a container yard. The second problem deals with the scheduling of RMGCs within a yard zone. This is not just a special case of the first problem, because inter-crane interferences have to be considered as well. For both problems two different models are presented: the job model and the workload model. Both models are different ways to interpret the work that has to be done by the cranes. In Figure 6, the problems are classified with respect to the physical abilities of the cranes and to the two different models. After introducing some general parameters and assumptions, that are identical in both models, the characteristics of the two different models are described and the mathematical models are developed. The objective of all the models below is to minimize the total unfinished workload at the end of each time period. This idea is also used in [2] and in [18] and seems to be a good measurement for the effectiveness of crane schedules.

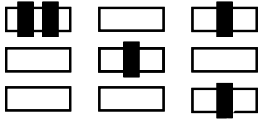

Models Cranes	Job Model	Workload Model
RTGC: 	Rubber Tired Job Problem (RTJP) see section 3.2.1	Rubber Tired Workload Problem (RTWP) see section 3.3.1
RMGC: 	Rail Mounted Job Problem (RMJP) see section 3.2.2	Rail Mounted Workload Problem (RMWP) see section 3.3.2

Figure 6: Problem classification

3.1 General parameters and assumptions

- K denotes the number of cranes available
- In all models, the cranes are regarded as identical machines
- Due to the physical size of the blocks and to the potential danger of crane collision, the maximum number of cranes that can work simultaneously in a block is bounded and denoted by \tilde{K}
- The planning horizon is divided into T small time periods
- The planning horizon starts at time $t = 1$
- Time period t denotes the time between time t and $t + 1$; as a consequence, the planning horizon ends at time $T + 1$
- One crane can do one unit of work in one time period
- Any crane movement can only start at the beginning of a time period

- Work can only start at the beginning of a time period
- L denotes the number of blocks
- τ_{lm} denotes the travel time of a crane from block l to block m
- It is assumed that the travel times are symmetric and satisfy the triangle inequality
- m_k denotes the initial crane location (a block or a slot) of crane k at the beginning of the planning horizon
- c_l denotes the initial number of cranes in block l at the beginning of period 1
- M represents a large number in the formulations. Although M could be replaced by an expression formulated with the used parameters in every formulation, M is used to keep the formulations simple.

3.2 The job models

The idea of the job model is based on scheduling theory. The work which has to be done by the cranes is modeled as jobs. Each job j has a release date r_j , a processing time p_j and a location l_j . The release date is the earliest time possible the work which is summarized by job j can be started by a crane. It is assumed that $r_j \leq T$ for $j = 1, \dots, J$. The processing time p_j is the number of time units that one crane needs to process job j completely. The location l_j or s_j indicates in which block or in which slot the work summarized by job j occurs. In the job model, it is assumed that every job is performed by a single crane and can not be interrupted if it has been started once and it can not be started more than once. In the following, the two problems mentioned above are modeled by using these parameters.

3.2.1 The job model for RTGC scheduling

One of the characteristics of RTGCs is that they can move freely among the blocks of the container yard, so no interference conditions have to be taken in consideration in this case. Further, it is assumed that a crane can always take the shortest path between two blocks. The RTGC scheduling problem, described as a job model, is referred to as the “rubber tired job problem” (RTJP). The following decision variables are used to formulate the RTJP:

$$x_{kjt} = \begin{cases} 1 & \text{if crane } k \text{ starts job } j \text{ at time } t \\ 0 & \text{else} \end{cases}$$

$$y_{klt} = \begin{cases} 1 & \text{if crane } k \text{ stays in block } l \text{ during period } t \\ 0 & \text{else} \end{cases}$$

Let $p_h(l)$ be the set of blocks that a yard crane located in block l can possibly be in at period $t - h$ and let $s_h(l)$ be the set of blocks that a yard crane located in block l can possibly be in at period $t + h$. Using these decision variables and sets, the RTJP_1 can be formulated as an integer program as follows:

$$\text{Min} \quad \sum_{j=1}^J \left(\left(\sum_{k=1}^K \sum_{t=1}^T \left(x_{kjt} \cdot \left((t-r_j)p_j + \sum_{h=1}^{\min(p_j; T-t+1)} (p_j-h) \right) \right) \right) + \left(1 - \sum_{k=1}^K \sum_{t=1}^T x_{kjt} \right) \cdot (T-r_j+1)p_j \right)$$

subject to

$$(1) \quad \sum_{k=1}^K \sum_{t=1}^T t \cdot x_{kjt} \geq r_j \cdot \sum_{k=1}^K \sum_{t=1}^T x_{kjt} \quad \text{for } j = 1, \dots, J$$

$$(2) \quad \sum_{k=1}^K \sum_{t=1}^T x_{kjt} \leq 1 \quad \text{for } j = 1, \dots, J$$

$$(3) \quad \sum_{\substack{i=1 \\ i \neq j}}^J \sum_{h=1}^{\min(p_j-1; T-t)} x_{k,i,t+h} \leq M(1-x_{kjt}) \quad \text{for } k = 1, \dots, K, j = 1, \dots, J: p_j > 1, \\ t = 1, \dots, T-1$$

$$(4) \quad \sum_{h=0}^{\min(p_j-1; T-t)} y_{k,l_j,t+h} \geq p_j \cdot x_{kjt} \quad \text{for } k = 1, \dots, K, j = 1, \dots, J, t = 1, \dots, T$$

$$(5a) \quad y_{klt} \leq \sum_{h=1}^{t-1} \sum_{m \in p_h(l)} y_{k,m,t-h} \quad \text{for } k = 1, \dots, K, l = 1, \dots, L, \\ t = 2, \dots, T$$

$$(5b) \quad y_{klt} \leq \sum_{h=1}^{T-t} \sum_{m \in s_h(l)} y_{k,m,t+h} \quad \text{for } k = 1, \dots, K, l = 1, \dots, L, \\ t = 1, \dots, T-1$$

$$(6) \quad \sum_{l=1}^L y_{klt} \leq 1 \quad \text{for } k = 1, \dots, K, t = 1, \dots, T$$

$$(7) \quad \sum_{k=1}^K y_{klt} \leq \tilde{K} \quad \text{for } l = 1, \dots, L, t = 1, \dots, T$$

$$(8) \quad x_{kjt} \in \{0,1\} \quad \text{for } k = 1, \dots, K, j = 1, \dots, J, t = 1, \dots, T$$

$$(9) \quad y_{klt} \in \{0,1\} \quad \text{for } k = 1, \dots, K, l = 1, \dots, L, \\ t = 1, \dots, T$$

As assumed above, the objective of this program is to minimize the remaining unfinished workload at the end of each time period. The unfinished workload is calculated for each job j and is summed up for all jobs. The unfinished workload for a job j is calculated as follows: If job j is started during the planning horizon, there exists exactly one x_{kjt} for all k and for all t which is equal to 1. If $x_{kjt} = 1$, then the t in the index is the starting time of job j . During the time periods between the release date and the starting time $(t-r_j)$, the remaining unfinished workload at the end of a time period equals the whole processing time of job j . When the job is started at t , the workload is reduced by one unit in each time period until the job is done or until the end of the planning horizon is reached. In this case, $\sum_{k=1}^K \sum_{t=1}^T x_{kjt} = 1$ and the second term vanishes. If job j is not started during the planning horizon, i.e. $x_{kjt} = 0$ for all k and all t ,

then the first term vanishes and the unfinished workload is counted by the second term ($\sum_{k=1}^K \sum_{t=1}^T x_{kjt}$ is equal to zero). Because job j is not started, the unfinished workload at the end of each period between the release date and the end of the planning horizon is the processing time of j .

Constraint (1) requires that job j can only be started after its release date. (2) guarantees that every job is only started once and only by one crane, (3) ensures that a crane can only start a new job if the old one has been completed. A crane also has to stay in the block when it processes a job in this block which is required by (4). (5a) and (5b) are crane movement constraints. (6) ensures that a crane can at most be in one block. (7) gives an upper bound for the number of cranes in a block.

RTJP_1 seems to be a rather complicated formulation of the RTJP. In the following, it is discussed how this formulation can be simplified.

One fact making RTJP_1 very complex is that it is not known whether or not the jobs are started and finished within the planning horizon. In order to find a simpler formulation, it is assumed that an upper bound π for the makespan of the jobs does exist. If this π is greater than T , then the planning horizon is extended beyond T up to π , while the length of the time periods remains constant. A method to find such a π is discussed in section 5.1. In a formulation, that uses such an extended planning horizon, it can be assumed that all jobs are completed within the planning horizon.

A planning horizon of π time periods allows to omit the second term of the objective function, because all jobs are started within the planning horizon. Therefore, the new objective function has the following form:

$$\text{Min} \quad \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} \left(x_{kjt} \cdot \left((t - r_j) p_j + \sum_{h=1}^{p_j} h \right) \right)$$

This function can be further simplified in the following way:

$$\begin{aligned} \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} \left(x_{kjt} \cdot \left((t - r_j) p_j + \sum_{h=1}^{p_j} h \right) \right) &= \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} (t - r_j) p_j + \sum_{j=1}^J \left(\sum_{k=1}^K \sum_{t=1}^{\pi} \left(x_{kjt} \cdot \sum_{h=1}^{p_j} h \right) \right) \\ &= \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} \cdot t \cdot p_j - \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} \cdot r_j \cdot p_j + \sum_{j=1}^J \sum_{h=1}^{p_j} h \\ &= \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} - \underbrace{\sum_{j=1}^J r_j \cdot p_j + \sum_{j=1}^J \sum_{h=1}^{p_j} h}_{const.} \end{aligned}$$

The last two terms can be omitted in the objective function, because they are constant. As a consequence, the new objective function has the following form:

$$\text{Min} \quad \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt}$$

This new objective function could also be interpreted as the minimization of the weighted sum of the starting times of the jobs, where the weights are given by the processing times of the jobs. It is an interesting result that a minimization of the remaining processing time at the

end of each time period is equivalent to the minimization of the weighted sum of the starting times. If the original objective function value is of interest, i.e. the remaining processing time at the end of each time period with a planning horizon of T time periods, then it can be calculated backward by the following procedure:

Procedure to calculate the remaining processing time:

(1) Calculate the remaining processing time for each job j :

a) If j is finished until T , i.e. $\sum_{t=1}^{T-p_j+1} \sum_{k=1}^K x_{kjt} = 1$, then the remaining processing time is

$$\sum_{t=1}^{T-p_j+1} \sum_{k=1}^K (x_{kjt} \cdot (t - r_j) \cdot p_j) + \sum_{h=1}^{p_j-1} h.$$

b) If j is started before T but finished after T , i.e. $\sum_{t=T-p_j+2}^T \sum_{k=1}^K x_{kjt} = 1$, then the remaining processing time is

$$\sum_{t=T-p_j+2}^T \sum_{k=1}^K \left(x_{kjt} \cdot \left((t - r_j) \cdot p_j + \sum_{h=1}^{T-t-1} p_j - h \right) \right)$$

c) If j is started after T , i.e. $\sum_{t=1}^T \sum_{k=1}^K x_{kjt} = 0$, then the remaining processing time is

$$(T - r_j + 1) \cdot p_j$$

(2) Sum up the remaining processing times of all jobs

A different problem in the formulation RTJP_1 could be caused by the sets $p_h(l)$ and $s_h(l)$. Those sets need to be determined for many different values of h and are hardly to handle in a solution procedure. To overcome these problems, the variables y_{klt} are replaced by the following variables:

$$y_{klmt} = \begin{cases} 1 & \text{if crane } k \text{ moves from block } l \text{ to block } m \text{ at the beginning of period } t \\ 0 & \text{else} \end{cases}$$

Using the variables y_{klmt} instead of y_{klt} , the RTJP can be formulated without using the sets $p_h(l)$ and $s_h(l)$.

A formulation RTJP_2 of the RTJP with planning horizon π , with the simplified objective function and with y_{klmt} as variables for the crane movement is presented below:

$$\text{Min} \quad \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt}$$

subject to

- $$\begin{aligned} (1) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \geq r_j && \text{for } j = 1, \dots, J \\ (2) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} = 1 && \text{for } j = 1, \dots, J \\ (3) \quad & \sum_{\substack{i=1 \\ i \neq j}}^J \sum_{h=1}^{p_j-1} x_{k,i,t+h} \leq M(1 - x_{kjt}) && \text{for } k = 1, \dots, K, j = 1, \dots, J: p_j > 1, \\ & && t = 1, \dots, \pi - p_j + 1 \\ (4) \quad & \sum_{h=0}^{p_j-1} y_{k,l_j,l_j,t+h} \geq p_j \cdot x_{kjt} && \text{for } k = 1, \dots, K, j = 1, \dots, J, \\ & && t = 1, \dots, \pi - p_j + 1 \\ (5) \quad & \sum_{m=1}^L y_{klmt} = \sum_{\substack{m=1 \\ m \neq l}}^L y_{k,m,l,t-\tau_{ml}} + y_{k,l,l,t-1} && \text{for } k = 1, \dots, K, l = 1, \dots, L, t = 2, \dots, \pi \\ (6) \quad & \sum_{k=1}^K y_{kl t} \leq \tilde{K} && \text{for } l = 1, \dots, L, t = 1, \dots, \pi \\ (7) \quad & x_{kjt} \in \{0,1\} && \text{for } k = 1, \dots, K, j = 1, \dots, J, \\ & && t = 1, \dots, \pi \\ (8) \quad & y_{klmt} \in \{0,1\} && \text{for } k = 1, \dots, K, l, m = 1, \dots, L, \\ & && t = 1, \dots, \pi \end{aligned}$$

The constraints (1) – (4) correspond in their interpretation to the constraints (1) – (4) of RTJP_1. (5) is the constraint for the possible crane movements and (6) corresponds in the meaning to (6) of RTJP_1. RTJP_2 seems to be simpler than RTJP_1 and is used to develop a solution procedure for the RTJP.

3.2.2 The job model for RMGC scheduling

RMGCs are more limited in their possibilities to move than RTGCs. They can only move along the rails on which they are mounted in a row of blocks within a yard zone. The RMGC scheduling problem described as a job model is referred to as the “rail mounted job problem” (RMJP). Although the formulation of the RMJP is based on the formulation of the RTJP, some differences have to be considered: Because of the fact that several RMGCs are mounted on the same rail, the model has to be extended by inter-crane interference constraints to avoid crane collisions. It seems to be very difficult to model inter-crane interference with the variables y_{klmt} , because too many cases have to be considered. Therefore, the crane movement is modeled by using the variables y_{klt} and the sets $p_h(l)$ and $s_h(l)$. Because of the fact that

3. Mathematical models

the cranes can only move within a row of blocks, the sets $p_h(l)$ and $s_h(l)$ are easy to determine and they even do not occur explicitly in the formulation.

In the formulation of the RMJP, the whole yard zone is divided into slots $s = 1, \dots, S$ in the following way: Every block is divided into \tilde{K} slots and it is assumed that the travel time between two adjacent slots is equal to 1. The space between two blocks is divided into τ virtual slots, where τ is the travel time of a crane between two adjacent blocks, so $S = L \cdot \tilde{K} + (L-1) \cdot \tau$. In each slot, there can only be one crane at a given time t .

The same decision variables as in RTJP_1 are used to formulate the RMJP. Further, the planning horizon is also extended as in RTJP_2 beyond T up to π , which is an upper bound for the makespan. In section 5.2, it is discussed how such an upper bound for the RMJP could be found. W.l.o.g. it is assumed that the cranes are numbered in such a way that $m_{k-1} < m_k$ holds for all $k = 2, \dots, K$. The RMJP can now be formulated as an integer program as follows:

$$\begin{aligned}
 \text{Min} \quad & \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \\
 \text{subject to} \quad & \\
 (1) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \geq r_j && \text{for } j = 1, \dots, J \\
 (2) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} = 1 && \text{for } j = 1, \dots, J \\
 (3) \quad & \sum_{\substack{i=1 \\ i \neq j}}^J \sum_{h=1}^{p_j-1} x_{k,i,t+h} \leq M(1-x_{kjt}) && \text{for } k = 1, \dots, K, j = 1, \dots, J: p_j > 1, \\
 & && t = 1, \dots, \pi - p_j + 1 \\
 (4) \quad & \sum_{h=0}^{p_j-1} y_{k,s_j,t+h} \geq p_j \cdot x_{kjt} && \text{for } k = 1, \dots, K, j = 1, \dots, J, \\
 & && t = 1, \dots, \pi - p_j + 1 \\
 (5a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} && \text{for } k = 1, \dots, K, s = 2, \dots, S-1, \\
 & && t = 2, \dots, \pi \\
 (5b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} && \text{for } t = 2, \dots, \pi \\
 (5c) \quad & y_{KS t} \leq y_{K,S,t-1} + y_{K,S-1,t-1} && \text{for } t = 2, \dots, \pi \\
 (6) \quad & \sum_{k=1}^K y_{kst} \leq 1 && \text{for } s = 1, \dots, S, t = 1, \dots, \pi \\
 (7) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} && \text{for } k = 2, \dots, K, s = 1, \dots, S, \\
 & && t = 1, \dots, \pi \\
 (8) \quad & x_{kjt} \in \{0,1\} && \text{for } k = 1, \dots, K, j = 1, \dots, J, \\
 & && t = 1, \dots, \pi \\
 (9) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, \\
 & && t = 1, \dots, \pi
 \end{aligned}$$

The objective function remains the same as in RTJP_2. (1) – (4) correspond in the meaning with (1) – (4) of RTJP_1. (5a) – (5c) are restrictions on crane movements. They correspond to (5a) and (5b) of RTJP_1, but they look simpler because the sets $p_h(l)$ and $s_h(l)$ are known and can be expressed explicitly. Constraint (6) ensures that there can be at most one crane in one slot at a given time. (7) is the inter-crane interference constraint which avoids crane collisions on the rail.

3.3 The workload models

The idea of the workload model is taken from [2]. In this model, the work which has to be done by the cranes is regarded as an amount of workload measured in time units. It is assumed that it is known how much time one crane needs to perform the work which occurs in a specific block l at a specific time t . The work in this model is given by the parameters w_{lt} . The workload occurring in block l at time t is measured in time periods. Using the workload model, the same problems as in section 3.2 can be modeled.

3.3.1 The workload model for RTGC scheduling

The RTGC scheduling problem described as a workload model is referred to as the “rubber tired workload problem” (RTWP), which is already developed in [2]. By using the variables

x_{lmt} = number of cranes moving from block l to block m at the beginning of period t

u_{lt} = unfinished workload in block l at the end of period t

the RTWP can be formulated as a mixed integer problem as follows:

$$\text{Min} \quad \sum_{t=1}^T \sum_{l=1}^L u_{lt}$$

subject to

$$(1) \quad \sum_{m=1}^L x_{lmt} \geq \sum_{\substack{m=1 \\ m \neq l}}^L x_{m,l,t-\tau_{ml}} + x_{l,l,t-1} \quad \text{for } l = 1, \dots, L, \quad t = 2, \dots, T$$

$$(2) \quad \sum_{l=1}^L x_{lm1} = c_l \quad \text{for } l = 1, \dots, L$$

$$(3) \quad x_{llt} \leq \tilde{K} \quad \text{for } l = 1, \dots, L, \quad t = 1, \dots, T$$

$$(4) \quad u_{l,t-1} + w_{lt} - u_{lt} \leq x_{llt} \quad \text{for } l = 1, \dots, L, \quad t = 1, \dots, T$$

$$(5) \quad u_{lt} \geq 0 \quad \text{for } l = 1, \dots, L, \quad t = 1, \dots, T$$

$$(6) \quad x_{lmt} \geq 0, \text{ integer} \quad \text{for } l, m = 1, \dots, L, \quad t = 1, \dots, T$$

For explanations see [2].

3.3.2 The workload model for RMGC scheduling

The RMGC scheduling problem described as a workload model is referred to as the “rail mounted workload problem” (RMWP). The yard zone is also divided into slots as in the formulation of the RMJP. The problem can be formulated as a combination of the RTWP and the crane movement constraints of the RMJP in the following way:

$$\begin{aligned} \text{Min} \quad & \sum_{t=1}^T \sum_{s=1}^S u_{st} \\ \text{subject to} \quad & \\ (1a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} \quad \text{for } k = 1, \dots, K, \quad s = 2, \dots, S-1, \\ & \quad \quad \quad t = 2, \dots, T \\ (1b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} \quad \text{for } t = 2, \dots, T \\ (1c) \quad & y_{KS t} \leq y_{K,S,t-1} + y_{K,S-1,t-1} \quad \text{for } t = 2, \dots, T \\ (2) \quad & \sum_{k=1}^K y_{kst} \leq 1 \quad \text{for } s = 1, \dots, S, \quad t = 1, \dots, T \\ (3) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} \quad \text{for } k = 2, \dots, K, \quad s = 1, \dots, S, \quad t = 1, \dots, T \\ (4) \quad & u_{s,t-1} + w_{st} - u_{st} \leq \sum_{k=1}^K y_{kst} \quad \text{for } s = 1, \dots, S, \quad t = 1, \dots, T \\ (5) \quad & u_{st} \geq 0 \quad \text{for } s = 1, \dots, S, \quad t = 1, \dots, T \\ (6) \quad & y_{kst} \in \{0,1\} \quad \text{for } k = 1, \dots, K, \quad s = 1, \dots, S, \quad t = 1, \dots, T \end{aligned}$$

The objective function is taken from the formulation of the RTWP and (1a) - (3) correspond to the crane movement constraints (5a) – (6) of the RMJP. (4) ensures that the workload that is done in a slot s during period t , is smaller than or equal to the number of cranes in the slot, i.e. it is smaller than one, because there can be at most one crane in a slot during period t .

4. Complexity

Theorem 4.1: *The RTJP is NP-hard in the strong sense.*

Proof: The proof is done by reducing the symmetric traveling salesman problem (STSP) satisfying the triangle inequality, which is known to be NP-hard in the strong sense, to the RTJP. Any instance of the STSP has a set of C cities, a positive integer D and distances $d(c_l, c_m) \in \mathbb{Z}^+$ between each pair of cities c_l, c_m . The distances are symmetric and satisfy the triangle inequality. The instance asks whether or not there exists a traveling salesman tour with length D or less. The associated instance of the RTJP asks whether or not there exists a crane schedule, in which the weighted sum of starting times of all jobs, where the weights are given by the processing times, is smaller than or equal to E . For any instance of the STSP with $C > 1$, a corresponding RTJP can be constructed in the following way:

$$\begin{aligned}
 J &= C \\
 L &= C \\
 \pi &= C(D+C) \\
 K &= 1 && (\tilde{K}=1) \\
 m_1 &= 1 \\
 \tau_{lm} &= d(c_l, c_m) && \text{for } l, m = 1, \dots, L \\
 r_1 &= D+C \\
 r_j &= 1 && \text{for } j = 2, \dots, C \\
 p_1 &= (C-1)(D+C) \\
 p_j &= 1 && \text{for } j = 2, \dots, C \\
 l_j &= c_j && \text{for } j = 1, \dots, C \\
 E &= (C-1)(D+C)(D+C+1)
 \end{aligned}$$

Depending on C , this construction can be done in polynomial time. In this instance of the RTJP, there is only one crane located in block 1 at the beginning and there is one job to do with processing time 1 in each of the blocks $2, \dots, C$. At time $D+C$, a job in block 1 with processing time $(C-1)(D+C)$ occurs. For these two instances, the following claim has to be shown.

Claim: For the given STSP instance there exists a traveling salesman tour with length D or less (i.e. the instance is a “yes”-instance) \Leftrightarrow
 For the given RTJP there exists a crane schedule such that the sum of the starting times weighted by the processing times is smaller or equal than E (i.e. the instance is a “yes”-instance)

Proof: “ \Rightarrow ”: Let $c_{l_1} - c_{l_2} - \dots - c_{l_C} - c_{l_1}$ be a traveling salesman tour for the given STSP with a total length of D or less and let w.l.o.g. $l_1 = 1$. Consider a schedule in which the crane travels among the blocks according to the sequence $l_1, l_2, \dots, l_C, l_1$, where it handles the respective job when it arrives at each block l_2, \dots, l_C . The crane starts moving at $t = 1$; its total travel time is at most D and it needs $C - 1$ time periods to do the jobs in blocks l_2, \dots, l_C . It will be back to block 1 at the beginning of period $D + C$ and will perform job 1 with the processing time $(C - 1)(D + C)$. The weighted sum of starting times after period $t = 1, 2, \dots, C(D + C)$ can be estimated as follows:

$$\begin{aligned}
 \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} &= p_1 \cdot \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} + \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \\
 &= (C - 1)(D + C) \cdot (D + C) \\
 &\quad + (1 + \tau_{l_1 l_2}) + (1 + \tau_{l_1 l_2} + \tau_{l_2 l_3} + 1) \\
 &\quad + \dots + (1 + \tau_{l_1 l_2} + \tau_{l_2 l_3} + \dots + \tau_{l_{C-1} l_C} + (C - 2)) \\
 &= (C - 1)(D + C)^2 + (C - 1) + \sum_{h=1}^{C-2} h \\
 &\quad + (C - 1)\tau_{l_1 l_2} + (C - 2)\tau_{l_2 l_3} + \dots + \tau_{l_{C-1} l_C} \\
 &\leq (C - 1)(D + C)^2 + \sum_{h=1}^{C-1} h \\
 &\quad + (C - 1)\tau_{l_1 l_2} + \dots + (C - 1)\tau_{l_{C-1} l_C} + (C - 1)\tau_{l_C l_1} \\
 &= (C - 1)(D + C)^2 + \frac{(C - 1)C}{2} + (C - 1)D \\
 &\leq (C - 1)((D + C)^2 + (D + C)) \\
 &= (C - 1)(D + C)(D + C + 1) \\
 &= E
 \end{aligned}$$

“ \Leftarrow ”: Suppose that there exists a solution for the RTJP such that the weighted sum of starting times is smaller than or equal to E . (*)

Assume that the crane comes back to block 1 later than period $D + C$ to process the job in block 1. Then

$$\begin{aligned}
 \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} &> p_1 \cdot \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} \\
 &\geq (C-1)(D+C)(D+C+1) \\
 &= E
 \end{aligned}$$

This would be a contradiction to (*). Therefore, the crane must come to block 1 no later than $D+C$.

Suppose now that there exists a job \tilde{j} in one of the blocks l_2, \dots, l_C , which is processed after job 1. Then

$$\begin{aligned}
 \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} &\geq p_1 \cdot \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} + \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k\tilde{j}t} \\
 &\geq p_1 \cdot \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} + \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} + p_1 \\
 &= p_1 \left(\sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} + 1 \right) + \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{k1t} \\
 &= (C-1)(D+C) \cdot ((D+C)+1) + (D+C) \\
 &> (C-1)(D+C)(D+C+1) \\
 &= E
 \end{aligned}$$

Therefore, the objective value of such a schedule would also exceed E . Hence, the crane has to do the jobs in the blocks l_2, \dots, l_C and comes back to block one at or before the period $D+C$ to process job 1, so the travel time of the crane will not exceed D . This implies that there is a solution to the STSP with a total length not greater than D . ■

The proof shows that the RTJP is NP-hard in the strong sense, even if $K = 1$. By using the same argumentation, it can be proved that the RMJP is also NP-hard in the strong sense because for $K = 1$, the RMJP can be formulated as a RTJP. In [2], it is proved that the RTWP is NP-hard in the strong sense by using the same idea, so it can be concluded that the RMWP is also NP-hard in the strong sense.

5. Upper bound for the makespan

5.1 Determine an upper bound π for the RTJP

The following procedure describes one way to find an upper bound π for the makespan for an instance of the RTJP. W.l.o.g. it is assumed that $r_j \leq r_{j+1}$ for $j = 1, \dots, J - 1$. It is also assumed that $J \geq K$. If $J < K$, then the same procedure can be used with slight changes.

Procedure to determine a π for the RTJP:

- (1) Assign all jobs j to a crane k in an arbitrary order, e.g. assume that crane k processes all jobs $\left\{ j = k + h \cdot K : h = 0, \dots, \left\lfloor \frac{J-k}{K} \right\rfloor \right\}$
- (2) Determine the makespan that each crane k needs to process the jobs assigned to it
- (3) The maximum makespan of all cranes can be used as π

The following algorithm shows how this procedure could be implemented.

Algorithm to determine a π for the RTJP:

Input: Set of jobs $\{1, \dots, J\}$
 Set of release dates $\{r_j : j = 1, \dots, J\}$
 Set of processing times $\{p_j : j = 1, \dots, J\}$
 Set of job locations $\{l_j : j = 1, \dots, J\}$
 Set of cranes $\{1, \dots, K\}$
 Set of initial crane locations $\{m_k : k = 1, \dots, K\}$

Output: Upper bound π for the makespan

Begin

Step1: For $k = 1$ to K do

$$t_k = \tau_{m_k, l_k} + p_k$$

For $h = 1$ to $\left\lfloor \frac{J-k}{K} \right\rfloor$ do

$$t_k = \max \left(t_k + \tau_{l_{k+(h-1)K}, l_{k+hK}} + p_{k+hK}; r_{k+hK} + p_{k+hK} \right)$$

5. Upper bound for the makespan

Step 2: $\pi = \max_{k=1,\dots,K} (t_k)$

End

5.2 Determine an upper bound π for the RMJP

The following procedure describes a way to find an upper bound π for the makespan for an instance of the RMJP.

Procedure to determine a π for the RMJP:

- (1) Assign all jobs located in the slots $1, \dots, m_2 - 1$ to crane 1
- (2) For $k = 2, \dots, K - 1$ assign all jobs located in the slots $m_k, \dots, m_{k+1} - 1$ to crane k
- (3) Assign all jobs located in the slots m_K, \dots, S to crane K
- (4) Determine the makespan that each crane k needs to process the jobs assigned to it
- (5) The maximum makespan of all cranes can be used as π

Algorithm to determine a π for the RMJP:

Input: Set of jobs $\{1, \dots, J\}$
Set of release dates $\{r_j : j = 1, \dots, J\}$
Set of processing times $\{p_j : j = 1, \dots, J\}$
Set of job locations $\{l_j : j = 1, \dots, J\}$
Set of cranes $\{1, \dots, K\}$
Set of initial crane locations $\{m_k : k = 1, \dots, K\}$

Output: Upper bound π for the makespan

Begin

Step 1: Set $n_k = m_k$ for $k = 1, \dots, K$

Step 2: Set $t_k = 0$ for $k = 1, \dots, K$

Step 3: For $j = 1$ to J do

If $l_j \leq m_2 - 1$ then $t_1 = \max(t_1 + \tau_{n_1, l_j} + p_j; r_j + p_j)$
 $n_1 = l_j$

Else if $l_j \geq m_K$ then $t_K = \max(t_K + \tau_{n_K, l_j} + p_j; r_j + p_j)$
 $n_K = l_j$

5. Upper bound for the makespan

Else For $k = 2$ to $K - 2$ do

 If $m_k \leq l_j \leq m_{k+1}$ then $t_k = \max(t_k + \tau_{n_k, l_j} + p_j; r_j + p_j)$
 $n_k = l_j$

Step 4: $\pi = \max_{k=1, \dots, K} (t_k)$

End

The algorithm determines for each job j in which “crane area” the job is located, i.e. which crane has to perform this job. Analogue to the algorithm above, the makespan that each crane needs to process the jobs assigned to it, is stored in the variable t_k . The variable n_k is needed to store the current position of crane k .

6. Solution procedures

6.1 Solution procedures for the RTJP

In section 4, it is proved that the RTJP is NP-hard in the strong sense. Therefore, the existence of an efficient solution procedure of this problem is very unlikely. In this section, two similar heuristics are proposed to get a good and feasible solutions for the RTJP. The heuristics will provide the schedule for the jobs in the form of a $J \times 2$ matrix S . Each row j contains the information on how job j is scheduled: the matrix entry s_{j1} is the number of the crane which performs job j in the schedule and the entry s_{j2} is the starting time of job j . This notation for the schedule is much more compact than the notation developed in section 3. Using this notation, it is also not necessary to determine an upper bound π for the makespan before a solution procedure can be started.

6.1.1 The Earliest Release Date - Heuristic

The objective of the RTJP is to minimize the weighted sum of starting times of the jobs, but a job can only be started at or after its release date, i.e. see constraint (1). The idea of the Earliest Release Date - Heuristic (ERD-Heuristic) is to schedule jobs with a small release date before jobs with a large release date, i.e. the jobs are scheduled according to their release dates, starting with the smallest release date.

W.l.o.g. it is assumed that the jobs are indexed in such a way that $r_j \leq r_{j+1}$ for $j = 1, \dots, J-1$. The idea of how to assign the jobs to the cranes is that every job should be performed by the crane which can be the first in the job location l_j . Two cases have to be considered to specify what is meant by “can be the first in l_j ”:

- Case 1: No job is assigned to crane k so far.
Then crane k can be in l_j after τ_{m_k, l_j} time units (that is the travel time of crane k from its initial crane location m_k to the job location l_j).
- Case 2: Jobs $j_1^{[k]}, \dots, j_h^{[k]}, h \in \{1, \dots, J-1\}$ with smaller release dates than r_j have already be assigned to crane k .
In this case, it is assumed that crane k has to perform the jobs $j_1^{[k]}, \dots, j_h^{[k]}$ before it can travel to block l_j . Therefore, the earliest time crane k can reach block l_j is the sum of the completion time of $j_h^{[k]}$ and the travel time between the location of $j_h^{[k]}$ and l_j .

The variables t_k and n_k are introduced in order to simplify the notation:

t_k is the earliest time crane k is available to process the next job. At the beginning of the scheduling process, $t_k = 0$ for all cranes. When a job is assigned to crane k , then t_k is updated,

The RTJP is formulated as an off-line problem. One advantage of the ERD-Heuristic is that it would also work for the corresponding on-line problem.

6.1.2 The Quotient-Heuristic

One disadvantage of the ERD-Heuristic is that it does not take the weights, i.e. the processing times of the jobs, into account. The Quotient-Heuristic tries to overcome this problem by considering the quotient $\frac{p_j}{r_j}$ and by scheduling the jobs in the order of these quotients, starting with the largest quotient. The job order is influenced by the release dates as well as by the weights of the jobs. The quotient $\frac{p_j}{r_j}$ could be regarded as a measurement for the “importance” of a job: The greater the weight p_j and the smaller the release date r_j , the greater is the importance of the job. With this interpretation, one can say that the Quotient-Heuristic schedules the jobs with respect to their importance, starting with the most important job. The rest of the procedure is equivalent to ERD-Heuristic.

Quotient-Heuristic:

Input: Set of jobs $\{1, \dots, J\}$
 Set of release dates $\{r_j : j = 1, \dots, J\}$
 Set of processing times $\{p_j : j = 1, \dots, J\}$
 Set of job locations $\{l_j : j = 1, \dots, J\}$
 Set of cranes $\{1, \dots, K\}$
 Set of initial crane locations $\{m_k : k = 1, \dots, K\}$

Output: Feasible schedule S for the RTJP

Begin

Step1: as in ERD-Heuristic

Step 2: Order the jobs such that $\frac{p_j}{r_j} \geq \frac{p_{j+1}}{r_{j+1}}$ for $j = 1, \dots, J - 1$

Step 3: as in ERD-Heuristic

End

6.2 Solution procedures for the RMJP

Ng proposes in [13] a two-phase scheduling heuristic for the RMJP: In the first phase, the yard zone is partitioned into K non-overlapping yard ranges. One crane is assigned to each of these yard ranges to handle all the jobs located in this range. A dynamic programming approach is used to determine an efficient partition. In the second phase, a job reassignment procedure is used to improve the schedule obtained in phase one. Although this heuristic is developed for the special case $p_j = p$ for $j = 1, \dots, J$, it can also be applied to the general case.

The ERD-Heuristic and the Quotient-Heuristic from section 6.1 can be modified in order to provide a solution for the RMJP. Step 1 and step 2 can remain unchanged; the only modification which has to be done is in step 3. There are at most two cranes which have to be taken in consideration to perform a job j : the crane which is the nearest to the job location l_j on the left hand side, and the crane which is the nearest to the job location l_j on the right hand side. From these two cranes, that one is chosen to perform job j which can be the earliest in l_j . Then Step 3 would have the following form:

For $j = 1$ to J do

Determine the crane, which is the nearest to l_j on the left hand side:

$$n_{k_l} = \max_{k: n_k \leq l_j} (n_k)$$

Determine the crane, which is the nearest to l_j on the right hand side:

$$n_{k_r} = \min_{k: n_k \geq l_j} (n_k)$$

Assign job j to crane \tilde{k} , where $\tilde{k} = \arg \min_{k \in \{k_l, k_r\}} (t_k + \tau_{n_k, l_j})$

6.3 Example

In this section, an example for the heuristics developed above is presented. The example should give an impression on how the ERD-Heuristic and the Quotient-Heuristic work.

The container yard in the example consists of nine blocks which are arranged in three rows with three blocks in each row. There are three jobs and two cranes are available to perform them. The positions of the cranes n_k and the locations of the jobs l_j are given by a vector (r, c) , where r indicates the row and c the column of the respective block in the container yard. It is assumed that the travel times of the cranes are rectilinear, i.e. the cranes can not move diagonally. The travel time between two adjacent blocks within the same row is one time period while the travel time between two adjacent blocks in the same column is two time periods. Figure 7 contains the job characteristics and Figure 8 illustrates the initial situation in the container yard. The transparent boxes in the blocks represent the jobs and the black boxes symbolize the cranes.

Jobs	r_j	p_j	l_j
1	1	5	(2,1)
2	2	5	(2,3)
3	15	5	(1,3)

Figure 7: Job characteristics

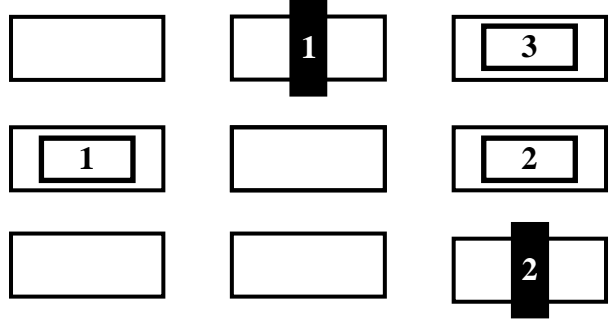


Figure 8: Initial situation

In this example, the order and the parameters of the jobs are chosen in such a way that $r_j \leq r_{j+1}$ and $\frac{p_j}{r_j} \geq \frac{p_{j+1}}{r_{j+1}}$ for $j = 1, \dots, 3$, so the jobs are in the right order for both heuristics.

This means that this example can be read as an example for the ERD-Heuristic as well as for the Quotient-Heuristic.

Step 1 is identical for both heuristics. The following setting is done in Step 1:

- $t_1 = t_2 = 0$
- $n_1 = (1, 2)$ and $n_2 = (3, 3)$

Step 2 can be omitted because the jobs are in the right order for both heuristics.

For the same reason, job 1 is the first job to be scheduled in step 3. It is easy to see that crane 1 needs one horizontal and one vertical move to reach the location of job 1, while crane 2 would need two horizontal and one vertical move. Therefore, job 1 is assigned to crane 1 ($s_{11} = 1$). Crane 1 starts its movement at $t = 1$. It needs one time period for the horizontal move and two time periods for the vertical move; so it reaches the location of job 1 at the beginning of time period 4. At this time, job 1 is already released, so job 1 is started at time period 4 ($s_{12} = 4$). Job 1 has a processing time of five time periods. Therefore, crane 1 is not available until time period 9 and t_1 is set equal to 9. The next job that has to be scheduled is job 2. Crane 1 would start from the location of job 1 and would therefore need two horizontal moves to come to the location of job 2, but it could not start before time period 9. Crane 1 would reach the location of job 2 at time period 11. Therefore, the second job is assigned to crane 2 ($s_{21} = 2$). This crane needs also two time periods for the vertical move to come to job 2, but it can start at period 1, so it reaches the location of job 2 at time period 3. The job is already released at this time, so the starting time is equal to 3 ($s_{22} = 3$). Now, crane 2 is not available until time period 8 and t_2 is set equal to 8. The last job which has to be done is job 3. Crane 1 could start at time period 9 from the location of job 1 and would need two horizontal moves and one vertical move to reach the location of job 3. Therefore, it could start job 3 at the beginning of period 13. Crane 2 can start to move at period 8 from the location of job 2 and needs two time periods for a vertical move to come to job 3, so it could start job 3 before crane 1, i.e. at the beginning of period 11, and job 3 is assigned to crane 2 ($s_{31} = 2$). Job 3 is not released until time period 15, i.e. $t_1 + \tau_{(2,3),(1,3)} = 8 + 2 = 10 < 15 = r_3$. Therefore, the starting time of job 3 is equal to 15. Figure 9 summarizes the schedule determined by the heuristics; Figure 10 illustrates the crane movements.

Jobs	Crane	Start
1	1	4
2	2	3
3	2	15

Figure 9: Schedule

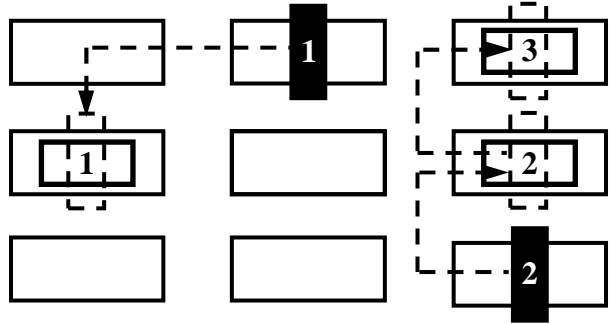


Figure 10: Illustration of crane movements

6.4 Idea for a Branch and Bound procedure

In this section, another idea is presented which could be used to develop an alternative solution procedure. In this thesis, the idea is not developed up to an algorithm, but it could be an inspiration for further research topics.

In section 3.2.1, the RTJP is formulated as an integer program. Two types of binary variables are used: x_{kjt} and y_{klmt} . The only relation between these variables is given by constraint (4).

In any other constraints, only one of the two types occurs. Therefore, it seems to be likely to work with Lagrangian relaxation to find a good lower bound for the RTJP. A Lagrangian relaxation could be used to design a branch and bound algorithm for the RTJP. In section 6.4.1, Lagrangian relaxation is used to divide the RTJP into two independent subproblems. In the sections 6.4.2 and 6.4.3, these subproblems are analyzed.

6.4.1 Lagrangian relaxation

Now, constraint (4) in the RTJP formulation of section 3.2.1 is placed in the objective function by Lagrangian relaxation with multipliers u_{kjt} for $k = 1, \dots, K$, $j = 1, \dots, J$, $t = 1, \dots, \pi - p_j + 1$. The relaxed problem has the following form:

$$L(\underline{u}) = \text{Min} \quad \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} - \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi - p_j + 1} u_{kjt} \cdot \left(\sum_{h=0}^{p_j-1} y_{k,l_j,l_j,t+h} - p_j \cdot x_{kjt} \right)$$

subject to

$$\begin{aligned} (1) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \geq r_j && \text{for } j = 1, \dots, J \\ (2) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} = 1 && \text{for } j = 1, \dots, J \\ (3) \quad & \sum_{i=1}^J \sum_{h=1}^{p_i-1} x_{k,i,t+h} \leq M(1 - x_{kjt}) && \text{for } k = 1, \dots, K, j = 1, \dots, J: p_j > 1, \\ & && t = 1, \dots, \pi - p_j + 1 \end{aligned}$$

6. Solution procedures

$$\begin{aligned}
 (4) \quad & \sum_{m=1}^L y_{klmt} = \sum_{\substack{m=1 \\ m \neq l}}^L y_{k,m,l,t-\tau_{ml}} + y_{k,l,l,t-1} & \text{for } k = 1, \dots, K, l = 1, \dots, L, t = 2, \dots, \pi \\
 (5) \quad & \sum_{k=1}^K y_{klit} \leq \tilde{K} & \text{for } l = 1, \dots, L, t = 1, \dots, \pi \\
 (6) \quad & x_{kjt} \in \{0,1\} & \text{for } k = 1, \dots, K, j = 1, \dots, J, \\
 & & t = 1, \dots, \pi \\
 (7) \quad & y_{klmt} \in \{0,1\} & \text{for } k = 1, \dots, K, l, m = 1, \dots, L, \\
 & & t = 1, \dots, \pi
 \end{aligned}$$

The relaxed problem can be decomposed into two independent subproblems, such that

$$L(\underline{u}) = L_1(\underline{u}) + L_2(\underline{u})$$

where

$$L_1(\underline{u}) = \text{Min} \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} + \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi-p_j+1} u_{kjt} \cdot p_j \cdot x_{kjt}$$

subject to

$$\begin{aligned}
 (1) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \geq r_j & \text{for } j = 1, \dots, J \\
 (2) \quad & \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} = 1 & \text{for } j = 1, \dots, J \\
 (3) \quad & \sum_{\substack{i=1 \\ i \neq j}}^J \sum_{h=1}^{p_j-1} x_{k,i,t+h} \leq M(1-x_{kjt}) & \text{for } k = 1, \dots, K, j = 1, \dots, J: p_j > 1, \\
 & & t = 1, \dots, \pi - p_j + 1 \\
 (4) \quad & x_{kjt} \in \{0,1\} & \text{for } k = 1, \dots, K, j = 1, \dots, J, t = 1, \dots, \pi
 \end{aligned}$$

and

$$L_2(\underline{u}) = \text{Min} \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi-p_j+1} -u_{kjt} \cdot \sum_{h=0}^{p_j-1} y_{k,l_j,l_j,t+h}$$

subject to

$$\begin{aligned}
 (1) \quad & \sum_{m=1}^L y_{klmt} = \sum_{\substack{m=1 \\ m \neq l}}^L y_{k,m,l,t-\tau_{ml}} + y_{k,l,l,t-1} & \text{for } k = 1, \dots, K, l = 1, \dots, L, t = 2, \dots, \pi \\
 (2) \quad & \sum_{k=1}^K y_{klit} \leq \tilde{K} & \text{for } l = 1, \dots, L, t = 1, \dots, \pi \\
 (3) \quad & y_{klmt} \in \{0,1\} & \text{for } k = 1, \dots, K, l, m = 1, \dots, L, \\
 & & t = 1, \dots, \pi
 \end{aligned}$$

6.4.2 The subproblem $L_1(\underline{u})$

Theorem 6.1: $L_1(\underline{u})$ is NP-hard in the strong sense.

Proof: Consider an instance of $L_1(\underline{u})$ with $\underline{u} = \underline{0}$. Then the objective function of $L_1(\underline{u})$ has the following form:

$$\text{Min} \sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt}$$

Now, the constant term $\sum_{j=1}^J p_j^2$ is added to the objective function. Then the function can be modified as follows:

$$\sum_{j=1}^J p_j \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} + \sum_{j=1}^J p_j^2 = \sum_{j=1}^J p_j \cdot \left(\sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} + p_j \right)$$

This objective function together with the constraints of $L_1(u)$ form the problem $\tilde{L}_1(u)$:

$$\tilde{L}_1(u) = \text{Min} \sum_{j=1}^J p_j \cdot \left(\sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} + p_j \right)$$

subject to

$$(1) \quad \sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} \geq r_j \quad \text{for } j = 1, \dots, J$$

$$(2) \quad \sum_{k=1}^K \sum_{t=1}^{\pi} x_{kjt} = 1 \quad \text{for } j = 1, \dots, J$$

$$(3) \quad \sum_{\substack{i=1 \\ i \neq j}}^J \sum_{h=1}^{p_j-1} x_{k,i,t+h} \leq M(1 - x_{kjt}) \quad \text{for } k = 1, \dots, K,$$

$$j = 1, \dots, J: p_j > 1, \\ t = 1, \dots, \pi - p_j + 1$$

$$(4) \quad x_{kjt} \in \{0, 1\} \quad \text{for } k = 1, \dots, K, j = 1, \dots, J, \\ t = 1, \dots, \pi$$

$\tilde{L}_1(\underline{u})$ is equivalent to $L_1(\underline{u})$, because the problems only differ in a constant term in the objective function. Obviously, $\tilde{L}_1(\underline{u})$ is also equivalent to the following scheduling problem:

$P \mid r_j \mid \sum w_j C_j$: Schedule J jobs with release dates r_j on P identical machines in order to minimize the sum of weighted completion times.

The cranes are P identical machines. By definition of x_{kjt} , $\sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt}$ is the starting time of job j and, therefore, $\sum_{k=1}^K \sum_{t=1}^{\pi} t \cdot x_{kjt} + p_j$ is the completion time C_j of job j . The weights w_j are given by the processing times p_j .

This problem is known to be NP-hard in the strong sense, e.g. see [16].

This proves that even the special case $L_1(\underline{0})$ is NP-hard in the strong sense and, therefore, $L_1(\underline{u})$ is NP-hard in the strong sense as well. ■

The theorem implies that it is hard to find the optimal objective value of $L_1(\underline{u})$, but a lower bound could be determined by an appropriate procedure.

6.4.3 The subproblem $L_2(\underline{u})$

The objective function of $L_2(\underline{u})$ can be modified in the following way:

$$\begin{aligned}
 & \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi-p_j+1} -u_{kjt} \cdot \sum_{h=0}^{p_j-1} y_{k,l_j,l_j,t+h} \\
 &= \sum_{j=1}^J \sum_{k=1}^K \left(-u_{kj1} \cdot (y_{k,l_j,l_j,1} + \dots + y_{k,l_j,l_j,p_j}) - u_{kj2} \cdot (y_{k,l_j,l_j,2} + \dots + y_{k,l_j,l_j,p_j+1}) \right. \\
 & \quad \left. - \dots - u_{k,j,\pi-p_j+1} \cdot (y_{k,l_j,l_j,\pi-p_j+1} + \dots + y_{k,l_j,l_j,\pi}) \right) \\
 &= \sum_{j=1}^J \sum_{k=1}^K \left(-u_{kj1} \cdot y_{k,l_j,l_j,1} - (u_{kj1} + u_{kj2}) \cdot y_{k,l_j,l_j,2} \right. \\
 & \quad \left. - \dots - (u_{kj1} + \dots + u_{k,j,p_j}) \cdot y_{k,l_j,l_j,p_j} - (u_{kj2} + \dots + u_{k,j,p_j+1}) \cdot y_{k,l_j,l_j,p_j+1} \right. \\
 & \quad \left. - \dots - (u_{k,j,\pi-2,p_j+1} + \dots + u_{k,j,\pi-p_j+1}) \cdot y_{k,l_j,l_j,\pi-p_j+1} \right. \\
 & \quad \left. - (u_{k,j,\pi-2,p_j+2} + \dots + u_{k,j,\pi-p_j+1}) \cdot y_{k,l_j,l_j,\pi-p_j+2} \right. \\
 & \quad \left. - \dots - u_{k,j,\pi-p_j+1} \cdot y_{k,l_j,l_j,\pi} \right) \\
 &= \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} \left(\sum_{h=\max(1,t-p_j+1)}^{\min(t,\pi-p_j+1)} u_{kjh} \right) \cdot y_{k,l_j,l_j,t}
 \end{aligned}$$

6. Solution procedures

With $a_{jt} = \sum_{h=\max(1;t-p_j+1)}^{\min(t;\pi-p_j+1)} u_{kjh}$, the objective function can be formulated as $\sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} a_{jt} \cdot y_{k,l_j,l_j,t}$.

With $b_{lt} = \sum_{j:l_j=l} a_{jt}$, the subproblem $L_2(\underline{u})$ can be formulated as follows:

$$L_2(\underline{u}) = \text{Min} \sum_{l=1}^L \sum_{k=1}^K \sum_{t=1}^{\pi} b_{lt} \cdot y_{k,l,l,t}$$

subject to

$$\begin{aligned} (1) \quad & \sum_{m=1}^L y_{klmt} = \sum_{\substack{m=1 \\ m \neq l}}^L y_{k,m,l,t-\tau_{ml}} + y_{k,l,l,t-1} & \text{for } k = 1, \dots, K, l = 1, \dots, L, t = 2, \dots, \pi \\ (2) \quad & \sum_{k=1}^K y_{klmt} \leq \tilde{K} & \text{for } l = 1, \dots, L, t = 1, \dots, \pi \\ (3) \quad & y_{klmt} \in \{0, 1\} & \text{for } k = 1, \dots, K, l, m = 1, \dots, L, \\ & & t = 1, \dots, \pi \end{aligned}$$

Note that b_{lt} is a function of the Lagrangian multiplier \underline{u} , i.e. $b_{lt} = b_{lt}(\underline{u})$. In order to solve $L_2(\underline{u})$, the following problem is considered:

$$\tilde{L}_2(\underline{u}) = \text{Min} \sum_{l=1}^L \sum_{t=1}^{\pi} b_{lt} \cdot z_{llt}$$

subject to

$$\begin{aligned} (1) \quad & \sum_{m=1}^L z_{lmt} = \sum_{\substack{m=1 \\ m \neq l}}^L z_{m,l,t-\tau_{ml}} + z_{l,l,t-1} & \text{for } l = 1, \dots, L, t = 2, \dots, \pi \\ (2) \quad & \sum_{m=1}^L z_{lmt} = c_l & \text{for } l = 1, \dots, L \\ (2) \quad & z_{llt} \leq \tilde{K} & \text{for } l = 1, \dots, L, t = 1, \dots, \pi \\ (3) \quad & z_{lmt} \geq 0, \text{ integer} & \text{for } l, m = 1, \dots, L, t = 1, \dots, \pi \end{aligned}$$

where z_{lmt} is the number of cranes leaving block l to travel to block m at the beginning of period t . $L_2(\underline{u})$ and $\tilde{L}_2(\underline{u})$ describe nearly the same problem. The difference is that in $L_2(\underline{u})$, the movement of every single crane is considered while in $\tilde{L}_2(\underline{u})$, the cranes which move from the same block l to the same block m at the same time, are “summarized” in the variables z_{lmt} . Now it is shown, how $\tilde{L}_2(\underline{u})$ can be solved efficiently and how an optimal solution of $\tilde{L}_2(\underline{u})$ can easily be transferred to an optimal solution of $L_2(\underline{u})$:

Theorem 6.2: *Problem $\tilde{L}_2(\underline{u})$ is a minimum cost network flow problem.*

Proof: The following proof is already done in [2]. It is repeated here to introduce the notation for the network which is used in further steps of the solution procedure.

Let $G = (V, A)$ be a directed graph with the set of vertices

$$V = \{v_{lt} : l = 1, \dots, L; t = 1, \dots, \pi + 1\} \cup \{\tilde{v}\}$$

and the set of arcs

$$\begin{aligned} A = & \left\{ (v_{lt}, v_{m, t+\tau_{lm}}) : l, m = 1, \dots, L; l \neq m; t = 1, \dots, \pi - \tau_{lm} + 1 \right\} \\ & \cup \left\{ (v_{lt}, v_{l, t+1}) : l = 1, \dots, L; t = 1, \dots, \pi \right\} \\ & \cup \left\{ (v_{l, \pi+1}, \tilde{v}) : l = 1, \dots, L \right\}. \end{aligned}$$

Vertex v_{lt} represents block l at the beginning of period t . The vertex $v_{l, \pi+1}$ represents block l at the end of the last period. The nodes v_{lt} are the sources of the network with a supply of c_l , while \tilde{v} is the sink of the network with a demand of $\sum_{l=1}^L c_l$. The arcs between the vertices represent the possible crane movements. The arcs $\{(v_{lt}, v_{l, t+1}) : l = 1, \dots, L; t = 1, \dots, \pi\}$ have unit costs $-u \cdot b_{lt}$ and a capacity of \tilde{K} . All other arcs have zero costs and infinite capacity. It is obvious that $\tilde{L}_2(u)$ is equivalent to the minimum cost network flow problem (with multiple source nodes) in the network G . ■

The theorem implies that $\tilde{L}_2(\underline{u})$ can be solved efficiently. The following procedure describes how an optimal solution \underline{z}^* of $\tilde{L}_2(\underline{u})$ can be transformed into an optimal solution \underline{y}^* of $L_2(\underline{u})$.

Procedure to transform \underline{z}^* into \underline{y}^* :

- (1) Construct a directed graph $G = (V, A)$ with the same set of vertices and the same set of arcs as in the network constructed in the proof of Theorem 6.2.
- (2) The capacity of each arc is set equal to the value of the flow in \underline{z}^* on this arc, i.e. the capacity of the arcs $\{(v_{lt}, v_{m, t+\tau_{lm}}) : l, m = 1, \dots, L; l \neq m; t = 1, \dots, \pi - \tau_{lm} + 1\}$ is set equal to z_{lm}^* , the capacity of the arcs $\{(v_{lt}, v_{l, t+1}) : l = 1, \dots, L; t = 1, \dots, \pi\}$ is set equal to z_{lt}^* , and the capacity of the arcs $\{(v_{l, \pi+1}, \tilde{v}) : l = 1, \dots, L\}$ is set equal to $z_{l, \tilde{v}, \pi+1}^*$.

(3) For $k = 1$ to K do

- a) Determine a path from m_k to \tilde{v}
- b) Set each variable y_{klmt} corresponding to this path equal to 1
- c) Reduce the capacity of each arc occurring in the path by 1

(4) All y_{klmt} which are not set equal to 1 in step 3 are set equal to 0.

Theorem 6.3: *This procedure transforms an optimal solution \underline{z}^* of $\tilde{L}_2(\underline{u})$ into an optimal solution \underline{y}^* of $L_2(\underline{u})$.*

Proof: Define

$C_k^I(v_{lt}) =$ Sum of capacities of all arcs $\{(v, v_{lt}) : v \in V, (v, v_{lt}) \in A\}$ in iteration k
 $(C_k^I(v_{lt}))$ could be regarded as the “incoming capacity” of vertex v_{lt}

$C_k^O(v_{lt}) =$ Sum of capacities of all arcs $\{(v_{lt}, v) : v \in V, (v_{lt}, v) \in A\}$ in iteration k
 $(C_k^O(v_{lt}))$ could be regarded as the “outgoing capacity” of vertex v_{lt}

$C_0^I(v_{lt})$ and $C_0^O(v_{lt})$ are the corresponding values at the beginning.

Assume that there exists a \tilde{k} such that there exists no path from $m_{\tilde{k}}$ to \tilde{v} in iteration \tilde{k} .

\Rightarrow There exists a v_{lt} such that there exists a path from $m_{\tilde{k}}$ to v_{lt} but no path from v_{lt} to $v_{m,t+1}$ for all $m = 1, \dots, L$

$\Rightarrow C_{\tilde{k}}^I(v_{lt}) > 0$ and $C_{\tilde{k}}^O(v_{lt}) = 0$

$\Rightarrow C_{\tilde{k}}^I(v_{lt}) > C_{\tilde{k}}^O(v_{lt})$

In each iteration k , $C_k^I(v_{lt})$ and $C_k^O(v_{lt})$ are reduced by the same value (They are reduced by 1 if v_{lt} is on the path determined or they are both not reduced if v_{lt} is not on the path).

$\Rightarrow C_0^I(v_{lt}) > C_0^O(v_{lt})$

$$\Rightarrow \sum_{\substack{m=1 \\ m \neq l}}^L z_{m,l,t-\tau_{ml}}^* + z_{l,l,t-1}^* > \sum_{m=1}^L z_{lmt}^*$$

This is a contradiction to the feasibility of \underline{z}^* for $\tilde{L}_2(\underline{u})$, so for each $k = 1, \dots, K$, a path from m_k to \tilde{v} can be determined. This implies that the solution \underline{y}^* generated by the procedure, satisfies condition (1) of $L_2(\underline{u})$ for $k = 1, \dots, K$, $l = 1, \dots, L$, $t = 2, \dots, \pi$.

$\sum_{k=1}^K y_{klt}^*$ is smaller than or equal to the capacity of the arc $(v_{lt}, v_{l,t+1})$ at the beginning, because the capacity is reduced by 1 for each crane using this arc.

$$\Rightarrow \sum_{k=1}^K y_{klt}^* \leq z_{lt}^*$$

$z_{lt}^* \leq \tilde{K}$ because \underline{z}^* is feasible for $\tilde{L}_2(\underline{u})$

$$\Rightarrow \sum_{k=1}^K y_{klt}^* \leq \tilde{K}$$

This means that \underline{y}^* satisfies condition (2) of $L_2(\underline{u})$ for $l = 1, \dots, L$, $t = 1, \dots, \pi$.

Either y_{klmt} is set equal to 1 in step 3.b) or it is set equal to 0 in step 4, so condition (3) of $L_2(\underline{u})$ is also satisfied for $k = 1, \dots, K$, $l, m = 1, \dots, L$, $t = 1, \dots, \pi$.

Therefore, \underline{y}^* is feasible for $L_2(\underline{u})$.

Assume now that \underline{y}^* is not optimal for $L_2(\underline{u})$.

Then there exists a solution \underline{y}^{**} such that

$$\sum_{l=1}^L \sum_{k=1}^K \sum_{t=1}^{\pi} b_{lt} \cdot y_{k,l,l,t}^{**} < \sum_{l=1}^L \sum_{k=1}^K \sum_{t=1}^{\pi} b_{lt} \cdot y_{k,l,l,t}^*$$

Then define

$$z_{lmt}^{**} = \sum_{k=1}^K y_{klt}^{**} \quad \text{for } l, m = 1, \dots, L, \quad t = 1, \dots, \pi$$

It is easy to check that \underline{z}^{**} is feasible for $\tilde{L}_2(\underline{u})$ and

$$\begin{aligned} \sum_{l=1}^L \sum_{t=1}^{\pi} b_{lt} \cdot z_{lmt}^{**} &= \sum_{l=1}^L \sum_{k=1}^K \sum_{t=1}^{\pi} b_{lt} \cdot y_{k,l,l,t}^{**} \\ &< \sum_{l=1}^L \sum_{k=1}^K \sum_{t=1}^{\pi} b_{lt} \cdot y_{k,l,l,t}^* \\ &= \sum_{l=1}^L \sum_{t=1}^{\pi} b_{lt} \sum_{k=1}^K y_{klt}^* \end{aligned}$$

$$\leq \sum_{j=1}^J \sum_{k=1}^K \sum_{t=1}^{\pi} -u \cdot b_{lt} \cdot z_{lt}^*$$

This is a contradiction to the optimality of \underline{z}^* for $\tilde{L}_2(\underline{u})$. Therefore, \underline{y}^* generated by the procedure is optimal for $L_2(\underline{u})$. ■

6.5 Solution procedures for the RTWP

Cheung et al. propose in [2] two efficient solution procedures for the RTWP: the Lagrangian Decomposition Method and the Successive Piecewise-Linear Approximation Method. For details see [2].

6.6 Solution procedures for the RMWP

The RMWP is strongly related to the RTWP. The RMWP is not discussed in [2], but the solution procedures for the RTWP presented in [2] can be modified to solve the RMWP. In the following two sections, the necessary modifications of the procedures are discussed. Some parts of the solution procedures are copied from [2]. This is done for several reasons: The notation introduced so far can be used. It is also easier to follow the solution procedure and to realize where modifications are necessary and where new ideas come in.

6.6.1 The Lagrangian Decomposition Method for the RMWP

The first step of the Lagrangian Decomposition Method is the decomposition of the RMWP, which is equivalent to section 2.1 in [2]:

The variable z_{st} for $s = 1, \dots, S, t = 1, \dots, T$ is introduced, which is the number of cranes in slot s during time period t . Then the RMWP can be formulated as follows:

$$\text{Min} \quad \sum_{t=1}^T \sum_{s=1}^S u_{st}$$

subject to

$$(1a) \quad y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} \quad \text{for } k = 1, \dots, K, s = 2, \dots, S-1, \\ t = 2, \dots, T$$

$$(1b) \quad y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} \quad \text{for } t = 2, \dots, T$$

$$(1c) \quad y_{KS t} \leq y_{K,S,t-1} + y_{K,S-1,t-1} \quad \text{for } t = 2, \dots, T$$

$$(2) \quad \sum_{k=1}^K y_{kst} \leq 1 \quad \text{for } s = 1, \dots, S, t = 1, \dots, T$$

$$(3) \quad \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} \quad \text{for } k = 2, \dots, K, s = 1, \dots, S, t = 1, \dots, T$$

6. Solution procedures

$$\begin{aligned}
 (4) \quad & z_{st} = \sum_{k=1}^K y_{kst} && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (5) \quad & u_{s,t-1} + w_{st} - u_{st} \leq z_{st} && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (6) \quad & u_{st} \geq 0 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (7) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\
 (8) \quad & z_{st} \in \{0,1\} && \text{for } s = 1, \dots, S, t = 1, \dots, T
 \end{aligned}$$

Now, constraint (4) is relaxed and placed in the objective function with Lagrangian multipliers α_{st} . The relaxed problem has the following form:

$$L(\alpha) = \text{Min} \quad \sum_{t=1}^T \sum_{s=1}^S \left(u_{st} + \alpha_{st} \cdot \left(z_{st} - \sum_{k=1}^K y_{kst} \right) \right)$$

subject to

$$\begin{aligned}
 (1a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} && \text{for } k = 1, \dots, K, s = 2, \dots, S-1, \\
 & && t = 2, \dots, T \\
 (1b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} && \text{for } t = 2, \dots, T \\
 (1c) \quad & y_{KSt} \leq y_{K,S,t-1} + y_{K,S-1,t-1} && \text{for } t = 2, \dots, T \\
 (2) \quad & \sum_{k=1}^K y_{kst} \leq 1 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (3) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} && \text{for } k = 2, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\
 (4) \quad & u_{s,t-1} + w_{st} - u_{st} \leq z_{st} && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (5) \quad & u_{st} \geq 0 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (6) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\
 (7) \quad & z_{st} \in \{0,1\} && \text{for } s = 1, \dots, S, t = 1, \dots, T
 \end{aligned}$$

The relaxed problem $L(\alpha)$ can be decomposed into two independent subproblems. One of them can be further separated into S smaller subproblems. The decomposition can be done as follows:

$$L(\alpha) = L'(\alpha) + \sum_{s=1}^S L_s''(\alpha),$$

where

$$L'(\alpha) = \text{Min} \quad \sum_{t=1}^T \sum_{s=1}^S \sum_{k=1}^K -\alpha_{st} \cdot y_{kst}$$

subject to

$$\begin{aligned}
 (1a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} && \text{for } k = 1, \dots, K, s = 2, \dots, S-1, \\
 & && t = 2, \dots, T
 \end{aligned}$$

6. Solution procedures

$$\begin{aligned}
(1b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} && \text{for } t = 2, \dots, T \\
(1c) \quad & y_{KSt} \leq y_{K,S,t-1} + y_{K,S-1,t-1} && \text{for } t = 2, \dots, T \\
(2) \quad & \sum_{k=1}^K y_{kst} \leq 1 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
(3) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} && \text{for } k = 2, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\
(4) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, t = 1, \dots, T
\end{aligned}$$

and

$$L'_s(\alpha) = \text{Min} \sum_{t=1}^T (u_{st} + \alpha_{st} \cdot z_{st})$$

subject to

$$\begin{aligned}
(4) \quad & u_{s,t-1} + w_{st} - u_{st} \leq z_{st} && \text{for } t = 2, \dots, T \\
(5) \quad & u_{st} \geq 0 && \text{for } t = 2, \dots, T \\
(6) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, t = 1, \dots, T \\
(7) \quad & z_{st} \in \{0,1\} && \text{for } t = 1, \dots, T
\end{aligned}$$

Theorem 6.4: *Problem $L'(\alpha)$ is a minimum cost network flow problem.*

Proof: The poof is similar to the proof in section 2.2 of [2], but this proof is done for the RTWP. For the RMWP, the network has to be designed in a different way, because other constraints, especially the inter-crane interference constraint, have to be taken into account.

Let $G = (V, A)$ be a directed graph with the set of vertices

$$V = \{v_{sst} : s = 1, \dots, S; t = 1, \dots, T+1\} \cup \{v_{s,s+1,t} : s = 1, \dots, S-1; t = 1, \dots, T\} \cup \{\tilde{v}\}$$

and the set of arcs

$$\begin{aligned}
A = & \left\{ (v_{sst}, v_{s,s,t+1}) : s = 1, \dots, S; t = 1, \dots, T \right\} \\
& \cup \left\{ (v_{sst}, v_{s,s+1,t}) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
& \cup \left\{ (v_{s,s+1,t}, v_{s,s,t+1}) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
& \cup \left\{ (v_{s,s+1,t}, v_{s+1,s+1,t+1}) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
& \cup \left\{ (v_{s,s,T+1}, \tilde{v}) : s = 1, \dots, S \right\}.
\end{aligned}$$

The node \tilde{v} has capacity K while all other nodes have capacity 1. The arcs $(v_{sst}, v_{s,s,t+1})$ have unit costs of $-\alpha_{st}$ and all other arcs have zero costs. Figure 11 illustrates how the network looks like for $S = 3$ and $T = 3$.

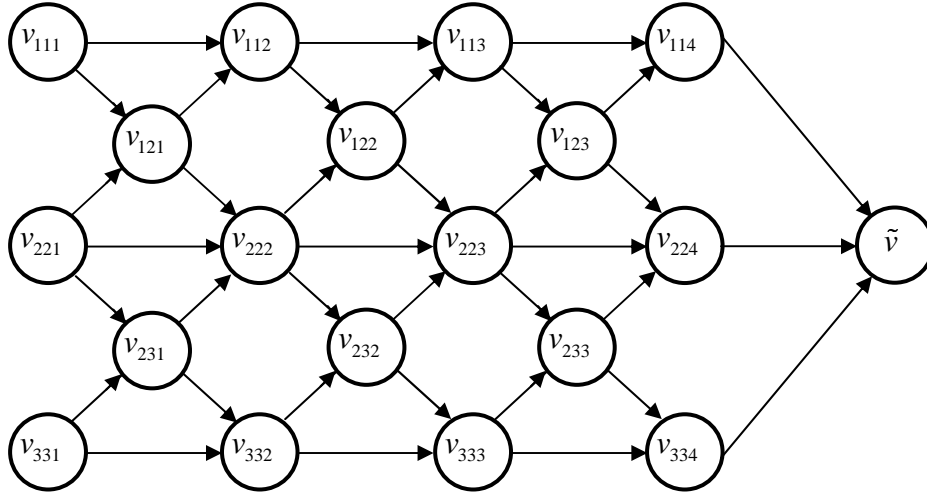


Figure 11: Illustration of the constructed network

The construction of the network is based on the same ideas as in the proof in [2]: Vertex v_{sst} represents slot s at the beginning of period t . Each node v_{sst} has capacity 1. This ensures that there can only be one crane in a certain slot at any time. The arcs $(v_{sst}, v_{s,s,t+1})$ are used by cranes which stay in a certain slot s during time period t . The nodes $v_{s,s+1,t}$ and all arcs connected to them ensure that two cranes can not pass each other. Assume there are two cranes in two adjacent blocks s and $s+1$. If the left crane wants to move to the right slot and the right crane wants to move to the left slot at the same time, then this is not possible in reality. It is also not possible in the network, because both cranes would have to pass the node $v_{s,s+1,t}$ in the network. This node has capacity 1 and therefore, a flow of two units can not pass this node at the same time. The supply of the nodes v_{ss1} is set equal to the number of cranes in a slot at the beginning of the planning horizon and the demand of the node \tilde{v} is set equal to K . The rest of the proof follows the argumentation of the proof in [2].

■

Theorem 6.4 implies that $L'(\alpha)$ can be solved efficiently by a minimum cost network flow algorithm. The node capacities could also be represented by arc capacities. This can be done by doubling the nodes and connecting them with arcs which have the original node capacity. Such a graph $G' = (V', A')$ would have the following set of vertices

$$\begin{aligned} V' = & \{v_{ss1}^o : s = 1, \dots, S\} \cup \{v_{ssT+1}^l : s = 1, \dots, S\} \\ & \cup \{v_{sst}^l : s = 1, \dots, S; t = 2, \dots, T\} \cup \{v_{sst}^o : s = 1, \dots, S; t = 2, \dots, T\} \\ & \cup \{v_{s,s+1,t}^l : s = 1, \dots, S-1; t = 1, \dots, T\} \cup \{v_{s,s+1,t}^o : s = 1, \dots, S-1; t = 1, \dots, T\} \cup \{\tilde{v}\} \end{aligned}$$

and the following set of arcs

$$\begin{aligned}
 A' = & \left\{ \left(v_{sst}^I, v_{sst}^O \right) : s = 1, \dots, S; t = 2, \dots, T \right\} \\
 & \cup \left\{ \left(v_{sst}^O, v_{s,s,t+1}^I \right) : s = 1, \dots, S; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{s,s+1,t}^I, v_{s,s+1,t}^O \right) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{sst}^O, v_{s,s+1,t}^I \right) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{s+1,s+1,t}^O, v_{s,s+1,t}^I \right) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{s,s+1,t}^O, v_{s,s,t+1}^I \right) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{s,s+1,t}^O, v_{s+1,s+1,t+1}^I \right) : s = 1, \dots, S-1; t = 1, \dots, T \right\} \\
 & \cup \left\{ \left(v_{ssT}^I, \tilde{v} \right) : s = 1, \dots, S \right\}.
 \end{aligned}$$

The solution of $L_s''(\alpha)$ and the construction of the overall solution procedure is analogue to the sections 2.3 and 2.4 of [2] and do not have to be repeated here.

6.6.2 The Successive Piecewise-Linear Approximation Method for the RMWP

The Successive Piecewise-Linear Approximation Method can also be applied to the RMWP. The procedure has to be modified in some parts because the crane movement constraints are different. Other parts can be simplified because the crane movement constraints are formulated with binary variables instead of integer variables. In the following, the Successive Piecewise-Linear Approximation Method for the RMWP is explained. The changes and modifications to section 3 of [2] are discussed in detail. Other Parts, which are similar to section 3 of [2] and which are necessary to understand the procedure, are presented in a more compact way.

In this approach, the RMWP is reformulated as a nonlinear programming problem with network flow constraints. The objective function is approximated by a function which is separable. The resulting problem is reformulated as a minimum cost network flow problem and a procedure to update the approximation is presented.

The RMWP can be rewritten as:

$$\text{Min} \quad f(\underline{y})$$

subject to

$$\begin{aligned}
 (1a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} && \text{for } k = 1, \dots, K, \quad s = 2, \dots, S-1, \\
 & && t = 2, \dots, T \\
 (1b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} && \text{for } t = 2, \dots, T \\
 (1c) \quad & y_{KS t} \leq y_{K,S,t-1} + y_{K,S-1,t-1} && \text{for } t = 2, \dots, T
 \end{aligned}$$

6. Solution procedures

$$\begin{aligned}
 (2) \quad & \sum_{k=1}^K y_{kst} \leq 1 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (3) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} && \text{for } k = 2, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\
 (4) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, t = 1, \dots, T
 \end{aligned}$$

where

$$f(\underline{y}) = \text{Min} \sum_{t=1}^T \sum_{s=1}^S u_{st}$$

subject to

$$\begin{aligned}
 (1) \quad & u_{s,t-1} + w_{st} - u_{st} \leq \sum_{k=1}^K y_{kst} && \text{for } s = 1, \dots, S, t = 1, \dots, T \\
 (2) \quad & u_{st} \geq 0 && \text{for } s = 1, \dots, S, t = 1, \dots, T
 \end{aligned}$$

$f(\underline{y})$ is separable in s . Let $y_{st} = \sum_{k=1}^K y_{kst}$ and $\underline{y}_s = (y_{s1}, \dots, y_{sT})$. Then $f(\underline{y})$ can be written as:

$$f(\underline{y}) = \sum_{s=1}^S f_s(\underline{y}_s)$$

where

$$f_s(\underline{y}_s) = \text{Min} \sum_{t=1}^T u_{st}$$

subject to

$$\begin{aligned}
 (1) \quad & u_{s,t-1} + w_{st} - u_{st} \leq y_{st} && \text{for } t = 1, \dots, T \\
 (2) \quad & u_{st} \geq 0 && \text{for } t = 1, \dots, T
 \end{aligned}$$

The optimal solution of $f_s(\underline{y}_s)$ can easily be determined by the following recursion:

$$u_{st}^* = \max \{u_{s,t-1}^* + w_{st} - y_{st}, 0\} \quad \text{for } t = 1, \dots, T$$

For the proof, see [2].

Now, the function $f_s(\underline{y}_s)$ is approximated by a function, which is separable in t :

Let $F = f(0)$. Then $f(\underline{y})$ is approximated by

$$f(\underline{y}) \approx F + \sum_{s=1}^S \sum_{t=1}^T \hat{f}_{st}(y_{st})$$

where each function $\hat{f}_{st}(y_{st})$ is a nonincreasing, linear function of $y_{st} \in [0,1]$, which satisfies the condition $\hat{f}_{st}(0)=0$. Let c_{st} be the slope of the line between the points $(0, \hat{f}_{st}(0))$ and $(1, \hat{f}_{st}(1))$. Then $\hat{f}_{st}(y_{st}) = c_{st} \cdot y_{st}$ and $f(\underline{y}) \approx F + \sum_{s=1}^S \sum_{t=1}^T c_{st} \cdot y_{st}$. Replacing y_{st} by $\sum_{k=1}^K y_{kst}$ leads to the following problem:

$$\text{Min} \quad F + \sum_{t=1}^T \sum_{s=1}^S \sum_{k=1}^K c_{st} \cdot y_{kst}$$

subject to

$$\begin{aligned} (1a) \quad & y_{kst} \leq y_{k,s-1,t-1} + y_{k,s,t-1} + y_{k,s+1,t-1} && \text{for } k = 1, \dots, K, s = 2, \dots, S-1, t = 2, \dots, T \\ (1b) \quad & y_{11t} \leq y_{1,1,t-1} + y_{1,2,t-1} && \text{for } t = 2, \dots, T \\ (1c) \quad & y_{KS t} \leq y_{K,S,t-1} + y_{K,S-1,t-1} && \text{for } t = 2, \dots, T \\ (2) \quad & \sum_{k=1}^K y_{kst} \leq 1 && \text{for } s = 1, \dots, S, t = 1, \dots, T \\ (3) \quad & \sum_{h=s}^S y_{k-1,h,t} \leq 1 - y_{kst} && \text{for } k = 2, \dots, K, s = 1, \dots, S, t = 1, \dots, T \\ (4) \quad & y_{kst} \in \{0,1\} && \text{for } k = 1, \dots, K, s = 1, \dots, S, t = 1, \dots, T \end{aligned}$$

Similar to the subproblem $L'(\alpha)$ in section 6.6.1, this is a minimum cost network flow problem and can be solved efficiently for any given set of objective function coefficients c_{st} . These coefficients are obtained as in section 3.3 of [2]: $f_s(\underline{y}_s)$ is solved for a given \underline{y}_s . Then, \underline{y}_s is changed in the t^{th} component and the problem is solved again. c_{st} is estimated by the difference of the objective function values of $y_{st}=0$ and $y_{st}=1$. The overall solution procedure can be constructed analogue to the SPLA Procedure in [2].

7. Lower bound

In this section, a procedure to determine a lower bound for the objective value of the job models is presented. This lower bound can be used to evaluate the performance of the heuristics developed in the sections 6.1.1 and 6.1.2. The basic idea for the procedure is the same as in section 6 of [13], but some problem specific changes and modifications have to be done. The first step is to find a simpler formulation for the problems.

Let

$$z_{jik} = \begin{cases} 1 & \text{if job } j \text{ is the } i^{\text{th}} \text{ job handled by crane } k \\ 0 & \text{else} \end{cases}$$

and $\underline{z} = (z_{111}, z_{112}, \dots, z_{JJK})$. Let $T_{jik}(\underline{z})$ be the starting time of job j done by crane k as the i^{th} job for a given \underline{z} . The problem of finding an optimal crane schedule can be formulated in the following way:

$$\text{Min} \quad \sum_{j=1}^J \sum_{i=1}^J \sum_{k=1}^K p_j \cdot z_{jik} \cdot T_{jik}(\underline{z})$$

subject to

$$\begin{aligned} (1) \quad & \sum_{i=1}^J \sum_{k=1}^K z_{jik} = 1 && \text{for } j = 1, \dots, J \\ (2) \quad & \sum_{j=1}^J z_{jik} \leq 1 && \text{for } i = 1, \dots, J, k = 1, \dots, K \\ (3) \quad & z_{jik} \in \{0, 1\} && \text{for } i, j = 1, \dots, J, k = 1, \dots, K \end{aligned}$$

This formulation holds for the RTJP and for the RMJP. The objective is the minimization of the weighted sum of starting times. Constraint (1) ensures that all jobs are performed and constraint (2) guarantees that each crane is only used once at a certain time. This integer program is equivalent to the assignment problem except that $T_{jik}(\underline{z})$ in the objective function is an unknown function of \underline{z} . The calculation of $T_{jik}(\underline{z})$ is as hard as solving the original problem. Therefore, it is NP-hard in the strong sense, but it can be used to determine a lower bound. Let \bar{T}_{jik} be a lower bound for the earliest time that a crane k can start job j as the i^{th} job in its schedule; obviously, \bar{T}_{jik} is a lower bound for $T_{jik}(\underline{z})$. If $T_{jik}(\underline{z})$ is replaced by \bar{T}_{jik} in the formulation above, then the following assignment problem is obtained.

$$\text{LB} = \text{Min} \quad \sum_{j=1}^J \sum_{i=1}^J \sum_{k=1}^K p_j \cdot \bar{T}_{jik} \cdot z_{jik}$$

subject to

7. Lower bound

$$\begin{aligned}
 (1) \quad & \sum_{i=1}^J \sum_{k=1}^K z_{jik} = 1 && \text{for } j = 1, \dots, J \\
 (2) \quad & \sum_{j=1}^J z_{jik} \leq 1 && \text{for } i = 1, \dots, J, \quad k = 1, \dots, K \\
 (3) \quad & z_{jik} \in \{0,1\} && \text{for } i, j = 1, \dots, J, \quad k = 1, \dots, K
 \end{aligned}$$

This problem is easy to solve, because it is obvious that the constraint matrix is totally unimodular and, therefore, the region of feasible solutions is an integral polyhedron. A solution for this problem is a lower bound for the objective value of the original problem. In the following, a procedure for finding a \bar{T}_{jik} is presented.

It follows from the definition of \bar{T}_{jik} that

$$\bar{T}_{j1k} = \max(\tau_{m_k, l_j} + 1; r_j) \quad \text{for } j = 1, \dots, J, \quad k = 1, \dots, K$$

and

$$\bar{T}_{jik} = \max\left(\min_{\substack{q \in \{1, \dots, J\} \\ q \neq j}} (\bar{T}_{q, i-1, k} + p_q + \tau_{l_q, l_j}); r_j\right) \quad \text{for } j = 1, \dots, J, \quad i = 2, \dots, J, \quad k = 1, \dots, K$$

Procedure to determine a lower bound:

- (1) Determine \bar{T}_{jik} for $i, j = 1, \dots, J, \quad k = 1, \dots, K$ by using the recursive equations above.
- (2) Solve Problem LB with these \bar{T}_{jik} and set the lower bound equal to the objective value of the solution of the problem LB.

8. Approximation bounds

If a problem can not be solved to optimality and heuristics have to be used to get feasible solutions, it is interesting to know, how well these heuristics approximate the optimal solution in the worst case, i.e. it is nice to have a kind of performance guarantee for the heuristics. In this section, such approximation bounds are determined for the RTJP and the RMJP. In both cases, the bounds are the same for the ERD-Heuristic and for the Quotient-Heuristic. It is obvious that an “absolute” approximation bound independent of the parameters for these heuristics does not exist, because it is easy to construct instances in which the gap between the optimal objective value and the solution determined by the heuristics depends on the values of the parameters. Therefore, the approximation bound determined in this section is a function of the following parameters of the instances:

- J
- K
- $r_j, \quad j = 1, \dots, J$
- $p_j, \quad j = 1, \dots, J$
- $\tau_{lm}, \quad l, m = 1, \dots, L$

The following notation is used to develop the approximation bounds:

- s_j = starting time of job j in the schedule determined by the heuristic
- $C^H = \sum_{j=1}^J p_j \cdot s_j$, objective value for the schedule determined by the heuristic
- $C^* = \sum_{j=1}^J p_j \cdot r_j$, lower bound for the optimal objective value
- $r_{\max} = \max_{j \in \{1, \dots, J\}} (r_j)$
- $p_{\max} = \max_{j \in \{1, \dots, J\}} (p_j)$
- $\tau_{\max} = \max_{l, m \in \{1, \dots, L\}} (\tau_{lm})$

In order to avoid misunderstandings in the notation, it is assumed w.l.o.g. that $J > K$. The argumentation would also hold (and would be simpler) if $J \leq K$. It is also assumed w.l.o.g. that the jobs are ordered according to the heuristic which is used, i.e. the jobs are ordered in such a way that $r_j \leq r_{j+1}$ for $j = 1, \dots, J - 1$ if the ERD-Heuristic is used or the jobs are

ordered such that $\frac{p_j}{r_j} \leq \frac{p_{j+1}}{r_{j+1}}$ for $j = 1, \dots, J - 1$ if the Quotient-Heuristic is used.

8.1 Approximation bound for the RTJP

Let

$$s_{\max}^n = \max_{j \in \{(n-1) \cdot K + 1, \dots, \min(J, n \cdot K)\}} (s_j) \quad \text{for } n = 1, \dots, \left\lceil \frac{J}{K} \right\rceil$$

Assume that each of the first K jobs would be done by a different crane and each crane would have to take the longest way possible to reach the location of its job. This worst-case-consideration provides the following upper bound for the starting time of the first K jobs in a schedule determined by the heuristic:

$$(1) \quad s_j \leq \max(\tau_{\max} + 1; r_j) \quad \text{for } j = 1, \dots, K$$

Assume further that each of the next K jobs, i.e. the jobs $K+1, \dots, 2 \cdot K$, would also be handled by a different crane and each crane would also have to take the longest possible way to reach the job location. But each crane has to finish its first job before it can travel to the location of the next job. An upper bound for the completion time of each of the first K jobs is $s_{\max}^1 + p_{\max}$. Therefore, each crane reaches its second job no later than $s_{\max}^1 + p_{\max} + \tau_{\max}$ and it follows that

$$(2) \quad s_j \leq \max(s_{\max}^1 + p_{\max} + \tau_{\max}; r_j) \quad \text{for } j = K + 1, \dots, 2 \cdot K$$

This argumentation can be repeated and it leads to the following upper bound

$$(3) \quad s_j \leq \max(s_{\max}^{n-1} + p_{\max} + \tau_{\max}; r_j) \quad \text{for } j = (n-1) \cdot K + 1, \dots, n \cdot K$$

Replace now r_j by r_{\max} in (1), (2) and (3) and define

$$\bar{s}_{\max}^1 = \max(\tau_{\max} + 1; r_{\max})$$

and

$$\bar{s}_{\max}^n = \max(\bar{s}_{\max}^{n-1} + p_{\max} + \tau_{\max}; r_{\max}) \quad \text{for } n = 2, \dots, \left\lceil \frac{J}{K} \right\rceil$$

It follows that

$$(4) \quad s_j \leq \bar{s}_{\max}^1 = \max(\tau_{\max} + 1; r_{\max}) \quad \text{for } j = 1, \dots, K$$

$$(5) \quad s_j \leq \bar{s}_{\max}^2 = \max(\bar{s}_{\max}^1 + p_{\max} + \tau_{\max}; r_{\max}) \quad \text{for } j = K+1, \dots, 2 \cdot K$$

\vdots

$$(6) \quad s_j \leq \bar{s}_{\max}^n = \max(\bar{s}_{\max}^{n-1} + p_{\max} + \tau_{\max}; r_{\max}) \quad \text{for } j = (n-1) \cdot K + 1, \dots, n \cdot K$$

8. Approximation bounds

Now, the second argument in the function $\max(\bullet; \bullet)$, i.e. the r_{\max} , can be omitted in all inequalities expressed by (5) and (6) because $\bar{s}_{\max}^{n-1} + p_{\max} + \tau > \bar{s}_{\max}^{n-1} \geq r_{\max}$ for $n = 2, \dots, \left\lceil \frac{J}{K} \right\rceil$.

Therefore, (5) and (6) can be simplified in the following way:

$$(7) \quad s_j \leq \bar{s}_{\max}^2 = \bar{s}_{\max}^1 + p_{\max} + \tau_{\max} \quad \text{for } j = K+1, \dots, 2 \cdot K$$

$$(8) \quad s_j \leq \bar{s}_{\max}^n = \bar{s}_{\max}^{n-1} + p_{\max} + \tau_{\max} \quad \text{for } j = (n-1) \cdot K + 1, \dots, n \cdot K$$

The inequalities (4), (7) and (8) can be summarized as follows:

$$(9) \quad s_j \leq \max(\tau_{\max} + 1; r_{\max}) + \left(\left\lceil \frac{j}{K} \right\rceil - 1 \right) \cdot (p_{\max} + \tau_{\max}) \quad \text{for } j = 1, \dots, J$$

Using inequality (9), the following upper bound for the weighted sum of starting times in a schedule determined by an heuristic can be obtained:

$$\begin{aligned} C^H &= \sum_{j=1}^J p_j \cdot s_j \\ &\leq \sum_{j=1}^J p_j \cdot \left(\max(\tau_{\max} + 1; r_{\max}) + \left(\left\lceil \frac{j}{K} \right\rceil - 1 \right) \cdot (p_{\max} + \tau_{\max}) \right) \\ &= \max(\tau_{\max} + 1; r_{\max}) \cdot \sum_{j=1}^J p_j + (p_{\max} + \tau) \cdot \sum_{j=1}^J \left(p_j \cdot \left(\left\lceil \frac{j}{K} \right\rceil - 1 \right) \right) \end{aligned}$$

Together with the lower bound C^* defined above, the following approximation bound is obtained:

$$C^H \leq C^* + \left(\max(\tau_{\max} + 1; r_{\max}) \cdot \sum_{j=1}^J p_j + (p_{\max} + \tau_{\max}) \cdot \sum_{j=1}^J \left(p_j \cdot \left(\left\lceil \frac{j}{K} \right\rceil - 1 \right) \right) - \sum_{j=1}^J p_j \cdot r_j \right)$$

Theorem 8.1: *This bound is tight for the Quotient-Heuristic.*

Proof: Consider the following instance of the RTJP:

- There are L blocks which are arranged in one row
- There is one crane which is initially located in block 1
- The travel time between two adjacent blocks is equal to 1 and, therefore, τ_{\max} is equal to $L - 1$
- There are two jobs with the following job characteristics:

Job	1	2
r_j	$L + 1$	1
p_j	$L + 2$	1
l_j	L	1

In such an instance with $J = 2$ and $K = 1$, $\max(\tau_{\max} + 1; r_{\max}) = r_{\max}$ and the upper bound for the weighted sum of starting times developed above has the following form:

$$\begin{aligned}
& \max(\tau_{\max} + 1; r_{\max}) \cdot (p_1 + p_2) + (p_{\max} + \tau_{\max}) \cdot \sum_{j=1}^2 \left(p_j \cdot \underbrace{\left(\left\lceil \frac{j}{1} \right\rceil - 1 \right)}_{=j-1} \right) \\
&= r_{\max} \cdot (p_1 + p_2) + (p_{\max} + \tau_{\max}) \cdot p_2 \\
&= p_1 \cdot r_{\max} + p_2 \cdot (r_{\max} + p_{\max} + \tau_{\max})
\end{aligned}$$

If the Quotient-Heuristic is applied, job 1 is handled before job 2 because $\frac{p_1}{r_1} = \frac{L+2}{L+1} > 1 = \frac{p_2}{r_2}$. Therefore, the crane has to travel to block L and reaches it at time $1 + (L-1) = L$. The crane starts job 1 at its release date $L+1$, which is equal to r_{\max} . Job 1 is finished at time $(L+1) + (L+2)$. The crane travels to block 1 in order to perform job 2. The travel time is $L-1$, so the second job is started at time $r_{\max} + p_{\max} + \tau_{\max}$. So if the Quotient-Heuristic is applied to such an instance, the objective function value reaches its upper bound.

If job 2 would be handled before job 1, both jobs could be started at their release dates. So there exists a feasible solution with an objective value equal to the lower bound $\sum_{j=1}^J p_j \cdot r_j$. This example shows that the approximation bound determined above is tight for the Quotient-Heuristic. ■

8.2 Approximation bound for the RMJP

For the RMJP, it can not be assumed that the first K jobs can be performed by K different cranes, because the crane movement constraints in the formulation in section 3.2.2 may not allow such an assignment. In the worst case, these jobs have to be done by only one crane. In this case, the problem can be regarded as a problem in which K is equal to 1. An instance of the RMJP with only one crane can be treated as an instance of the RTJP in which all blocks are arranged in one row. Therefore, the argumentation in section 8.1 can be repeated for the RMJP with K is equal to 1 and the following approximation bound for the RMJP is obtained:

$$C^H \leq C^* + \left(\max(\tau_{\max} + 1; r_{\max}) \cdot \sum_{j=1}^J p_j + (p_{\max} + \tau_{\max}) \cdot \sum_{j=1}^J p_j \cdot (j-1) - \sum_{j=1}^J p_j \cdot r_j \right)$$

By the same proof as in section 8.1, it can be shown that this bound is tight for the Quotient-Heuristic.

9. Computational experiments

In this section, the performance of the heuristics developed in section 6 is tested with randomly generated problem instances. The heuristics are coded in C++. The C++ program generates also a data file for AMPL. AMPL is used to calculate the lower bound described in section 7. For further details about the implementation see the CD attached to this thesis. All tests are run on an AMILO M7424 Notebook with an Intel Pentium M, FSB: 400 MHz processor. In section 9.1, it is described how the random problem instances are generated and in section 9.2, the results of the computational experiments are analyzed.

9.1 Problem generation

In this section, it is described how the problem instances are generated and how the parameters of the instances are set.

9.1.1 Problem instances for the RTJP

The RTJP is characterized by the number of blocks, the arrangement of the blocks, the number of time periods, the number of cranes available and the job characteristics. In every instance, there are four rows of blocks with five blocks in each row, so there is a total number of 20 blocks in the container yard. There are eight cranes available in the container yard. The number of cranes which can work simultaneously in a block is set equal to two. The initial location of each crane is determined randomly by a uniform distribution over the whole container yard. One time period is assumed to be three minutes and the problem instances are generated for a short time planning horizon of one hour and for a long time planning horizon of eight hours. Therefore, T is set equal to 20 in the first series and equal to 160 in a second series. It is assumed that the travel times of the cranes are rectilinear, i.e. the cranes can not move diagonally. In order to analyze the performance of the heuristics under different conditions, two settings for the travel times are considered: In a first setting, the travel time between two adjacent blocks in the same row is one time period, while the travel time between two adjacent blocks in the same column is two time periods. In a further test series, the travel time between two blocks in the same row is five time periods and the travel time between two blocks in the same column is ten time periods. Therefore, the travel effort is small with respect to the processing times of the jobs in the first series and it is large in the second series. The job parameters p_j, r_j and l_j are determined randomly by uniform distributions. The job location of each job is chosen randomly in the whole container yard, the release dates are distributed randomly over the whole planning horizon and the processing times lie between one and ten time periods. The number of jobs per hour is estimated as follows: One hour has 20 time periods and there are eight cranes available in the container yard. Therefore, at most $20 \cdot 8 = 160$ units of work can be done during one hour. An average job takes 5,5 time units. Therefore, a number of $\frac{160}{5,5} \approx 29,1$ jobs can be regarded as an

estimation for the capacity of the system. (This estimation does not consider the travel times of the cranes.) The heuristics are tested with 30 jobs per hour to simulate a busy system working on capacity and with 15 jobs per hour to simulate a system when there is not much work to do. With two planning horizons, two settings for the travel time and two possibilities

for the number of jobs per hour, there are eight different test cases. Each case is tested with five instances, so 40 tests are done for the RTJP.

9.1.2 Problem instances for the RMJP

The RMJP is also characterized by the number of blocks, the number of time periods, the number of cranes available and the job characteristics. In every instance, a row of ten blocks is considered. There are four cranes available in such a row to have the same crane-block-ratio as in the RTJP instances and \tilde{K} is also set equal to two. The initial location of the cranes is determined randomly by a uniform distribution over the row of container blocks. According to the ideas and the argumentation of section 3.2.2, the row of blocks has to be divided into slots: One block is divided into two slots, because $\tilde{K} = 2$ and the space between two blocks is represented by one slot. Therefore, each instance has a total number of $10 \cdot 2 + 9 = 29$ slots. A time period is assumed to be one minute, because the rail mounted cranes can move faster than the rubber tired cranes. In some papers on rail mounted cranes, the travel time of the cranes is even neglected. T is set equal to 60 and to 480 respectively to consider the same planning horizons as in the RTJP. Similar to the RTJP instances, two settings for the travel times are considered: In a first series, the travel time between two slots is equal to one and in a second series the travel time is set equal to ten. The job parameters p_j, r_j and l_j are also determined randomly by uniform distributions as in the instances of the RTJP. The processing times lie between 1 and 30 time periods, because a time period in the RMJP is shorter than a time period in the RTJP. With the same argumentation as for the RTJP instances, $\frac{60 \cdot 4}{15,5} \approx 15,5$ job per hour could be regarded as an upper bound for the capacity of the system. This estimate does also not consider the travel times. Furthermore, the movement of the cranes is much more restricted in the RMJP than in the RTJP. That's why the heuristics are tested with ten jobs per hour instead of 15 jobs to simulate a busy system and with five jobs per hour to model a system when there is not much work to do. As well as in section 9.1.1, there are eight different test cases. Each case is also tested with five instances, so 40 tests are done for the RMJP, too.

9.2 Results

In this section, the outcome of the computational experiments is discussed and analyzed. When the performance of a heuristic is analyzed, two properties of the heuristic are of interest: the computation time and the quality of the derived solution.

The computation time is the time a computer needs to calculate a solution based on the heuristic for a given problem instance. The tests show that a solution based on the heuristics can be calculated in real time or within several seconds for both problem types (RTJP and RMJP). So the computation time is not a critical factor for real world applications of the heuristics. For this reason, the computation time is not measured or further analyzed in this section.

The second property, i.e. the quality of the solution, has to be further specified: A heuristic can not deliver an optimal solution in general, but it should calculate a good solution with an objective value which is close to the optimal objective value. A good heuristic provides a solution with a small gap between the objective value of the solution and the optimal objective value. Unfortunately, the optimal objective value is not known for a given instance

and this gap can not be measured directly. Therefore, a lower bound for the optimal objective value is calculated (see section 7). The quotient of the objective value of the solution derived by the heuristic and the lower bound is used as a measurement for the quality of the heuristics. Figure 12 summarizes the tests for the RJTP and Figure 13 contains the test results for the RMJP. In both tables, the first three columns characterize the eight different test cases (see section 9.1). Each case is tested with five instances and for each instance, the quotient of the objective value derived by the heuristic and the lower bound is calculated. Column 4 and 5 contain the average of the five quotients of each test case for the respective heuristic. The heuristics are evaluated with the lower bound LB developed in section 7. In order to get an impression of the quality of this lower bound LB, it is tested with the trivial lower bound

$$LB^* = \sum_{j=1}^J p_j \cdot r_j \text{ using the same principles as for the evaluation of the heuristics. } LB^* \text{ is}$$

obviously a lower bound for the objective value, because the release date is by definition a lower bound for the starting time of a job. Column 6 contains the outcome of this evaluation. The original test data is stored on the CD attached to this thesis.

Computational Results for the RJTP					
Travel Time	Time Horizon	Jobs per hour	ERD / LB	Quotient / LB	LB / LB*
1 / 2	20	15	1,04	1,05	1,02
		30	1,48	1,43	1,05
	160	15	1,01	1,19	1,00
		30	1,50	1,51	1,00
5 / 10	20	15	1,27	1,23	1,30
		30	1,86	1,66	1,47
	160	15	1,39	1,42	1,01
		30	2,63	2,37	1,01

Figure 12: Computational results for the RJTP

In the following statements, the analysis of the computational results for the RJTP is summarized:

- The results show that both heuristics perform better for the first travel time setting. The heuristics provide very good results if the travel times are small with respect to the processing times of the jobs. If the travel times are large compared to the processing times, the results are still acceptable, but they are not as good as for the first setting of the travel times. This fact suggests, that one could try to improve the heuristics in such a way that the travel times are taken more into account than in the ERD-Heuristic or in the Quotient-Heuristic.

- It is interesting to see that the quotients do not differ a lot for the same travel time setting and the same number of jobs per hour, so the different time horizons do obviously not influence the performance of the heuristics. Although the difference becomes larger for the second travel time setting in a busy system (30 jobs per hour), it can be concluded for the RJTP that the time horizon seems not to be one of the key parameters for the performance of the heuristics.
- The level of work seems to be a relevant parameter: In every setting, the quotient of the objective value determined by the heuristic and LB increases considerably when the number of jobs per hour is increased. This fact suggests that the heuristics are better for systems in which there is not much work to do than for busy systems. A further reason for this effect could be that for instances representing a busy system, the lower bound LB is probably not as close to the optimal objective value as for the other systems. With more jobs per hour, there are more constraints making the feasible region smaller. This fact is not taken into account when the lower bound LB is determined.
- The quotients in the last column, which are all close to 1, express the relation between the lower bound LB from section 7 and the trivial lower bound LB*. The bounds are very close to each other, so the bound LB is not a very strong lower bound and should be improved. But when the heuristics are evaluated with a lower bound which is too small, the performance of the heuristic is obviously better than it is suggested by the values in Figure 12.

Computational Results for the RMTP					
Travel Time	Time Horizon	Jobs per hour	ERD / LB	Quotient / LB	LB / LB*
1	60	5	1,17	1,14	1,00
		10	1,55	1,19	1,04
	480	5	1,11	1,18	1,00
		10	1,38	1,39	1,00
10	60	5	1,15	1,12	2,47
		10	1,30	1,33	2,14
	480	5	1,44	1,48	1,03
		10	3,07	2,60	1,12

Figure 13: Computational results for the RMTP

The following statements contain an analysis of the computational results for the RMTP:

- While the travel time setting causes differences in the performance of the RJTP test series, the influence of the travel time setting in the RMJP is not really relevant. The performance is even better in some cases when the travel time is larger. One reason for this behaviour of the heuristics could be that the inter-crane interference constraints which are not relevant for the RTJP, condition the crane movement stronger than the travel times.
- As in the test series for the RTJP, it can be observed that the different time horizons do not have much influence on the performance of the heuristics. Although the difference becomes larger for the test series in the last row of Figure 13, the time horizon seems not to be an important parameter for the heuristics.
- When the number of jobs per hour is increased, the same effect can be observed as in the RJTP test series and the same argumentation holds.
- The quotient of the lower bounds LB and LB* is nearly equal to 1 for the test series with the first travel time setting and for the two test series with the second travel time setting and the large time horizon. This fact underlines the thesis that the lower bound LB is not very strong and should be improved. With the second travel time setting and the short planning horizon, the quotient of the lower bounds is greater than 2 and the heuristic solutions are acceptable for both levels of work. These are the only test series for which the calculation of the lower bound LB leads to a real improvement in the analysis of the results.

The computational results for both problem types (RTJP and RMJP) show in all test series that there is no difference in the performance of the ERD-Heuristic and the Quotient-Heuristic. Therefore, it is enough to run one of them for practical applications. If the problem instances have a special structure, it could make sense to prefer one of the heuristics, but for arbitrary instances, it does not matter which one is chosen.

10. Special Cases

In this section, two special cases of the problems presented in section 3 are discussed. It is not intended to find a new solution procedure for these special cases. They are just discussed to provide a better understanding for the problems.

10.1 $K = 1$

For a container yard in which only one crane is available, the problems RTJP and RMJP as well as the problems RTWP and RMWP are equivalent, because no inter-crane interference constraints have to be considered. For practical applications, the different travel time settings have to be taken into account: In the RTJP or in the RTWP, horizontal and vertical movements are possible, which differ in their travel effort (see the test cases in section 9). In the RMJP and in the RMWP, only horizontal moves within a yard zone are possible. But for a given travel time matrix $(\tau_{lm})_{\substack{l=1,\dots,L \\ m=1,\dots,L}}$, the mathematical formulation of the RTJP and the RMJP

coincide for $K = 1$ and the problems can be tackled by the same heuristics. This also holds for the RTWP and the RMWP. It is hard to find a solution procedure which provides an optimal solution for this special case, because the proof in section 4 and the proof in [2] are done for this case. Therefore, the problem is NP-hard in the strong sense for the job model and for the workload model even if K is equal to 1.

Ng and Mak propose in [14] a branch and bound algorithm for this problem with another objective function. They try to minimize the sum of the job waiting times.

The problem formulation developed in section 3 could be simplified for $K = 1$, but for only one crane, it seems to be reasonable to formulate the problem with other variables. An alternative formulation for the problem is presented below. The formulation is based on the ideas of the formulation in [14].

Let s_j be the starting time of job j and

$$x_{ij} = \begin{cases} 1 & \text{if job } i \text{ is handled before job } j \\ 0 & \text{else} \end{cases}$$

With these variables, the problem can be formulated as follows:

$$\text{Min} \quad \sum_{j=1}^J p_j \cdot s_j$$

subject to

- (1) $s_j \geq r_j$ for $j = 1, \dots, J$
- (2) $s_j - s_i \geq \tau_{i,j} + p_i - M \cdot (1 - x_{ij})$ for $i, j = 1, \dots, J: i \neq j$
- (3) $x_{ij} + x_{ji} = 1$ for $i, j = 1, \dots, J: i \neq j$
- (4) $x_{ij} \in \{0, 1\}$ for $i, j = 1, \dots, J: i \neq j$

$$(5) \quad s_j \geq 0, \text{ integer} \quad \text{for } j = 1, \dots, J$$

The objective is the minimization of the weighted sum of starting times of the jobs as in the models of section 3. Constraint (1) guarantees that every job is started at or after its release date. Constraint (2) ensures that a job j can only be started when its predecessor has been completed and the crane has reached the location of job j . Constraint (3) guarantees that either job i is processed before job j or job j is processed before job i . This formulation is simpler than the formulation of section 3 and can be handled better. The special case, in which only one crane is available, could be relevant in practice for small container terminal systems.

10.2 $p_j = 1$

When the jobs in a container yard are very similar and the processing times are small compared to the travel times of the cranes, a special case with small, constant processing times could be relevant in practice. $p_j = 1$ for all jobs is the easiest case of constant processing times. Moreover, this special case is also interesting from a theoretical point of view. When $p_j = 1$ for $j = 1, \dots, J$, each job is completed in the same time period in which it is started. Therefore, all constraints which ensure that a job can not be interrupted if it is started once can be neglected. That means that the job model and the workload model coincide in this special case. Therefore, each instance for the job model can be transferred into an instance for the workload model and visa versa without losing information. A job model instance can be transformed into a workload model instance by the following setting:

$$w_{lt} = \sum_{\substack{j: l_j = l \\ r_j = t}} p_j = \left| \{j \in J : l_j = l, r_j = t\} \right|$$

When a workload model instance should be transformed into an instance for the job model, then the jobs and the job characteristics are determined as follows:

For $l = 1$ to L do

For $t = 1$ to T do

Generate w_{lt} jobs with

- $p_j = 1$
- $r_j = t$
- $l_j = l$

Such an instance for the job model would have $J = \sum_{l=1}^L \sum_{t=1}^T w_{lt}$ jobs. While this transformation

is possible for an arbitrary instance, the transformation of a job model instance into a workload model instance without losing information is only possible in this special case. Besides the two models, the two heuristics developed in section 6.1 also coincide if $p_j = 1$ for

10. Special Cases

$j = 1, \dots, J$, because sorting the jobs such that $r_j \leq r_{j+1}$ for $j = 1, \dots, J - 1$ is equivalent to sorting the jobs such that $\frac{1}{r_j} \geq \frac{1}{r_{j+1}}$ for $j = 1, \dots, J - 1$.

The statements above hold for the RTJP and the RTWP as well as for the RMJP and the RMWP.

11. Conclusion

This diploma thesis examines logistic problems occurring in a container terminal. The thesis focuses on the scheduling of cranes handling containers in a port. Two problems are discussed in detail: the yard crane scheduling of rubber-tired gantry cranes (RMGC) which move freely among the container blocks, and the scheduling of rail-mounted gantry cranes (RMGC) which can only move within a yard zone. The problems are formulated as integer programs. For each of the two problems discussed, two models are presented: In one model, the crane tasks are interpreted as jobs with release times and processing times while in the other model, it is assumed that the tasks can be modeled as generic workload measured in crane minutes. It is shown that the problems are NP-hard in the strong sense. Heuristic solution procedures are developed and evaluated by numerical results. Further ideas which could lead to other solution procedures are presented and some interesting special cases are discussed.

The computational experiments show that both heuristics developed in this thesis could be improved. One drawback of both algorithms is that in each iteration only one job, i.e. the next job to be scheduled, is considered. It could be an interesting approach to consider more jobs simultaneously in order to minimize the travel effort of the cranes and to get more efficient solutions. Another outcome of the experiments is that it seems to be likely to find a better lower bound than the one presented in section 7. This would lead to a more accurate evaluation of the heuristics. Beside the problems discussed in this thesis, modified problem formulations could be an interesting field for further studies. It might be interesting for some practical applications to work with other objective functions or to take due dates of jobs into account. The scheduling of cranes within the container yard is only one part of the workflow in a container terminal. Further research on other logistic problems in an container yard, e.g. the loading and unloading of ships at the quayside, the scheduling of the internal trucks or all the location problems occurring in the container yard, would help to improve the workflow and the turnaround times of container terminals.

References

- [1] Chen, Hsieh (1999). A time-space network model for the berth allocation problem. 19th IFIP TC7 Conference on System Modeling and Optimization, Cambridge, UK
- [2] Cheung, Li, Lin (2002). Interblock crane deployment in container terminals. *Transportation Science*, Vol. 36, No.1, pp. 79 – 93
- [3] Daganzo (1989). The crane scheduling problem. *Transportation Research Part B*, Vol. 23, No. 3, pp. 159 – 175
- [4] Imai, Nagaiwa, Tat (1997). Efficient planning of berth allocation for container terminals in Asia. *Journal of Advanced Transportation* 31 (1), pp. 75 – 94
- [5] Imakita (1978). A techno-economic analysis of the port transport system. Praeger, New York
- [6] Kim, Kim (1997). A routing algorithm for a single transfer crane to load export containers onto a containership. *Computers and Industrial Engineering* 33, pp. 673 – 676
- [7] Kim, Kim (1999). An optimal routing algorithm for a transfer crane in port container terminals. *Transportation Science*, Vol. 33, No.1, pp. 17 – 33
- [8] Konzan, Preston (1999). Genetic algorithms to schedule container transfer at multimodal terminals. *International Transaction in Operational Research*, Vol. 6, pp. 311 – 329
- [9] Lai, Leung (1996). Analysis of yard crane deployment strategies in a container terminal. *Proceedings of the International Conference on Computers and Industrial Engineering 1996*, pp. 1187 – 1190
- [10] Lai, Shih (1992). A study of container berth allocation. *Journal of Advanced Transportation* 26 (1), pp. 45 – 60
- [11] Li, Cai, Lee (1998). Scheduling with multi-job-on-one-processor pattern. *IIE Transactions* 30, pp. 433 – 445
- [12] Lim (1998). The berth planning problem. *Operations Research Letters* 22, pp. 105 - 110
- [13] Ng (2005). Crane scheduling in container yards with inter-crane interference. *European Journal of Operational Research* 164, pp. 64 – 78
- [14] Ng, Mak (2005). Yard crane scheduling in port container terminals. *Applied Mathematical Modelling* 29, pp. 263 – 276
- [15] Peterkofsky, Daganzo (1990). A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B*, Vol. 24, No.3, pp. 159 – 172

- [16] Schulz, Skutella (2002). Scheduling unrelated machines by randomized rounding. *SIAM Journal on Discrete Mathematics*, Vol. 15, No. 4, pp. 450 – 469
- [17] Zhang (2000). Resource planning in container storage yard. Ph.D. Thesis. Department of Industrial Engineering and Engineering Management, Hong Kong University of Science and Technology, Hong Kong
- [18] Zhang, Wan, Liu, Linn (2002). Dynamic crane deployment in container storage yards. *Transportation Research Part B*, Vol. 36, No. 6, pp. 537 – 555

Picture Credits

Figure 1: <http://www.kline.co.jp/biz/terminal/ohi.html> (July 3rd, 2005)

Figure 2: <http://www.mpa.state.md.us/Intermodal/container.htm> (July 3rd, 2005)

Figure 3: <http://www.spedcont.com.pl/de/tkkrakow.htm> (July 3rd, 2005)

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Referenzen und Hilfsmittel erstellt zu haben. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Hornbach, August 2005

Peter Bohrer