Hoist Scheduling Problem

1 Introduction

This report presents the objectives and progress made in my Student Project on the *hoist scheduling problem* (HSP). The HSP originates from electroplating lines, where we have to design a fully automated system that is used to cover parts with a coat of metal. The parts are moved by one or more preprogrammed hoists between tanks containing various process specific solutions. In the first semester our study focused on the single-hoist cyclic HSP, which involved reviewing and analyzing relevant research articles on the topic, along with creating a solver and a visualization of the obtained soultions. As a next step, this study focuses on the multi-hoist setting, which is a significantly harder problem. Throughout the semester multiple research articles on this topic were examined and the proposed methods were implemented. All the reviewed papers had one thing in common: the solutions were not complete in the sense that the empty moves of the hoists were not specified, leading to incomplete solutions. As a contribution we designed a model, which creates the exact schedule for all hoists, given some sensible assumptions for the hoist movements.

2 Multi-hoist CHSP

Let us consider the problem described in [1]. In this setup, parts stored by carriers must undergo a sequence of chemical treatment. First the carriers are loaded into the system (see Figure 1) from a loading station (tank 0), then transported between the tanks in a given order (1 to n) and then moved back to the loading station. We are given empty-hoist and loaded-hoist movement times as input. The tanks are assumed to be indexed in the order of the soaking process, so the loaded hoist movements are only done between consecutive tanks. The carriers must be immersed into every tank for which minimum and maximum soaking times are given as input. No two carriers may occupy the same tank at the same time.

The amount of time between two successive introductions of carriers into the system (from the loading station) is called a *cycle*, which is a *cycle length* long time-span. This is called a simple cycle. It follows that the system must be in the same state after each cycle, meaning the same set of tanks are empty and full initially (tank 0 is always full). For simple cycles it is also true that during a cycle exactly one carrier is dropped into and is removed from each tank, because otherwise the same state could not be attained after the cycle.

The hoists are programmed to execute the same set of actions in each cycle. We assume that the hoists can perform 2 kinds of moves: empty-hoist moves from the tank they were used at to the tank they are next needed at and loaded moves between consecutive tanks. A hoist schedule consists of the hoist move sequence for each hoist and the associated start-times for the moves.

Our goal is to design a feasible hoist schedule, that is: the hoist movements are executable, collision free and for each tank the soaking times are within the specified bounds. Among these we want to find one that maximizes the throughput of carriers of the system. With the constraint that we want to find a simple cycle, this is equivalent to finding a schedule with minimum cycle length.

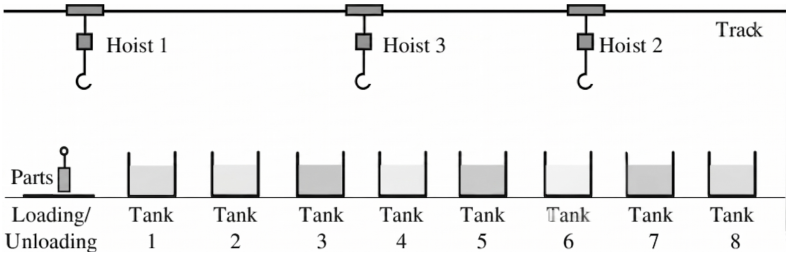


Figure 1: Illustration of the multi-hoist system

3 Solution method

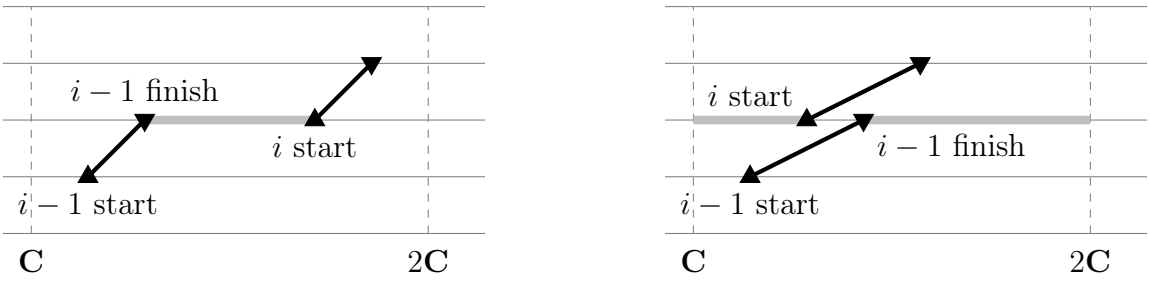
In the literature the most common way of finding an optimal schedule is by formulating and solving a mixed-integer program (MIP). Let us first consider the formulation, found in the article [3]. The formulation uses similar ideas as in the single-hoist case, which we covered last semester, but due to the increased complexity a lot of additional variables and constraints must be introduced. You can find all the necessary notation for the parameters and variables in Tables 1 and 2.

For instance let us look at a case where we have to introduce new variables (see Figure 2). From the single-hoist case the variables $y_{i,i-1}$, indicating whether move i starts before move $i-1$ are no longer sufficient to decide if the carrier is in tank i at the beginning of the cycle, which is crucial for writing up the soaking time constraints. This is so because if move $i-1$ starts before move i it might be the case that it finishes after the start of move i, which is possible if they are done by different hoists. This arrangement requires the carrier to be in tank i at the beginning of the cycle (the variable s_i equals 1). However if move $i-1$ finishes before move i then the carrier is not present in tank i initially (variable s_i equals 0).

The most interesting new constraints are the collision avoidance constraints. Let us look at one of them to see the underlying idea:

$$t_i + d_i + e_{i+1,j} - t_j \leq M\left(3 - y_{ij} - z_i^k - \sum_{h=k}^{K} z_j^h\right) \qquad \forall i,j \in N, j < i, k \in \mathcal{K} \quad (\star)$$

Figure 3 demonstrates a situation, when the above inequality is activated, namely when move j starts after move i and move j is carried out by a bigger indexed hoist than



(a) $y_{i,i-1} = 0$, $s_i = 0$ (b) $y_{i,i-1} = 0$, $s_i = 1$

Figure 2: The variables $y_{i,j}$ no longer express if a carrier is in a tank at the beginning

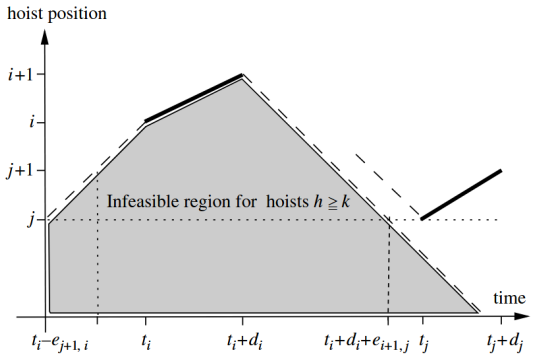


Figure 3: Illustration of constraint ($\star$) from [3].

move i. Fixing move i to hoist k, creates an unreachable area for hoists greater than k on the time-way diagram, since for any space-time point (s, t) of the shaded area, hoist k cannot be closer to tank 0 at time t than s due to the empty-hoist movement times and therefore no bigger indexed hoist can occupy that point. This constraint exactly bans move j from this area.

All in all, ($\star$) and other similar constraints restrict the feasible positions of each hoist at each point in time, so that it is possible to construct the hoist trajectories such that no collision occurs (for the simple case separation proof, see [3]). However, a formal procedure for constructing these hoist trajectories is missing, not only from [3], but from all reviewed papers.

4 Constructing the hoist trajectories

To illustrate the difficulties that may arise during the construction of the trajectories, Figure 4a shows the optimal solution given by the MIP for a problem instance from the literature called "black oxide 1" with 4 hoists. For example move 6 and move 7 are both done by the fourth hoist, but move 9, which is done by the first hoist, is between these moves. Therefore after hoist 4 completes move 6, it must go above tank 10 (even though it is not needed there) to evade collision with hoist 1 and then back to tank 7. In this section we are looking for a solution, where such evasion maneuvers are not allowed, resulting in a simpler schedule. We will see that for most of the instances found in the literature, this restriction does not increase the optimal cycle time, but allows us to precisely construct the schedule of each hoist.

In the new model if we consider two consecutive moves i and j of hoist k, only the following actions are allowed: waiting at tank i after the completion of move $i-1$, traveling empty directly to tank j and waiting at tank j before starting move j. Moreover no moves are allowed to go through the cycle, this is a sensible assumption since it would be difficult to initiate the system if the hoists were moving at the beginning. The new variables in Table 1 will help us to formulate the above mentioned requirements.

Table 1: Model Variables and Descriptions

Variable	**Description**
a_i	Arrival time to the starting tank of move i for $i \in N^0$,
b_i	Departure time from the destination tank of move i fo $i \in N^0$,
$x^k_{i,j}$	Indicator of move j being the immediate successor of move i on hoist k $\forall i, j \in N, \forall k \in \mathcal{K}$

Now we go through the new constraints, that apply the above mentioned restrictions. For the proper definition of the variables $x^k_{i,j}$, we define dummy moves s and t, representing the 0th move and $N+1$-th move.

$$\sum_{j \in N \cup t} x^k_{i,j} = z^k_i \qquad \forall i \in N, k \in \mathcal{K} \tag{1}$$

$$\sum_{j \in N \cup s} x^k_{j,i} = z^k_i \qquad \forall i \in N, k \in \mathcal{K} \tag{2}$$

$$\sum_{j \in N \cup t} x^k_{s,j} = 1 \qquad \forall k \in \mathcal{K} \tag{3}$$

$$\sum_{j \in N \cup s} x^k_{j,t} = 1 \qquad \forall k \in \mathcal{K} \tag{4}$$

$$x^k_{i,j} \leq y_{i,j} \qquad \forall i, j \in N, \forall k \in \mathcal{K} \tag{5}$$

Constraints (1)–(2) say, that for each move i and hoist k there is exactly one successor and one predecessor. Constraints (3)–(4) define the first and last move of each hoist and constraint (5) ensures that the x variables indeed follow the order of the moves previously

defined by the y variables.

$$a_i \leq t_i + M \sum_k x^k_{s,i} \qquad \forall i \in N^0 \tag{6}$$

$$b_i \geq t_i + d_i \qquad \forall i \in N^0 \tag{7}$$

$$b_i \leq C \qquad \forall i \in N^0 \tag{8}$$

$$a_i \leq C \qquad \forall i \in N^0 \tag{9}$$

Constraint (6) forces the arrival time to a move to be smaller than the start of the move, unless i is the first move of some hoist, because then it might be that we arrive to the ith tank at the end of the last cycle. Inequality (7) always forces the departure time to be greater than the finish of the corresponding move. This is a minor restriction, since it forbids inter-cycle waiting at the last tank, but it makes the formulation simpler. Constraints (8)–(9) just bound the variables by the cycle time.

$$a_j \geq b_i + e_{i+1,j} - M(1 - x^k_{i,j}) \qquad \forall i,j \in N, \forall k \in \mathcal{K} \tag{10}$$

$$a_j \leq b_i + e_{i+1,j} + M(1 - x^k_{i,j}) \qquad \forall i,j \in N, \forall k \in \mathcal{K} \tag{11}$$

Constraints (10)–(11) apply the rule that after departing from the previous move the hoist must go directly to the next tank.

$$b_i + e_{i+1,j} \leq a_j + M\left(2 - x^k_{s,j} - x^k_{i,t}\right) \qquad \forall i,j \in N, \forall k \in \mathcal{K} \setminus \{0\} \tag{12}$$

$$b_i + e_{i+1,j} \geq a_j - M\left(2 - x^k_{s,j} - x^k_{i,t}\right) \qquad \forall i,j \in N, \forall k \in \mathcal{K} \setminus \{0\} \tag{13}$$

Constraints (12)–(13) express the above mentioned rule for move pairs which are the last and first for a hoist.

$$x^h_{s,j} \leq 1 - x^k_{s,i} \qquad \forall k \in \mathcal{K}, \forall h < k, \forall i \in N, \forall j > i \tag{14}$$

$$x^h_{s,j} \leq 1 - x^k_{s,i} \qquad \forall k \in \mathcal{K}, \forall h > k, \forall i \in N, \forall j < i \tag{15}$$

Constraint (14) expresses that if hoist k starts with move i then a smaller h hoist cannot start with a bigger move j, while (15) is the counterpart of (14).

The following constraints are pretty technical, they ensure that there are no collisions at the end and at the beginning of the cycle.

$$z^h_l \leq 3 - \left(x^k_{s,i} + x^k_{j,t} + y_{j,l}\right) \qquad \forall i,j \in N, \forall k \in \mathcal{K}, \forall l < i, \forall h > k \tag{16}$$

$$z^h_l \leq 3 - \left(x^k_{s,i} + x^k_{j,t} + y_{j,l}\right) \qquad \forall i,j \in N, \forall k \in \mathcal{K}, \forall l > i, \forall h < k \tag{17}$$

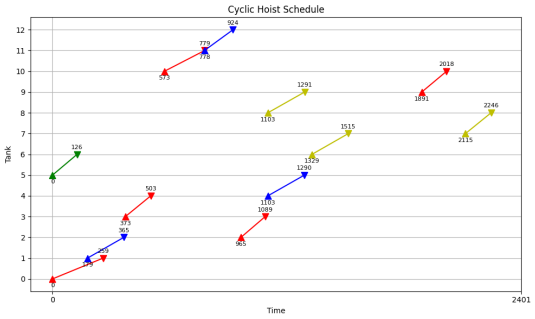
Constraint (16) expresses that if i is the first and j is the last move of hoist k and l is a smaller indexed move than i, such that it starts after move j, then l cannot be done by a hoist h, which is greater than k.

$$z^h_l \leq 2 - \left(x^k_{s,i} + y_{l,i}\right) \qquad \forall i \in N, \forall k \in \mathcal{K}, \forall l < i, \forall h > k \tag{18}$$

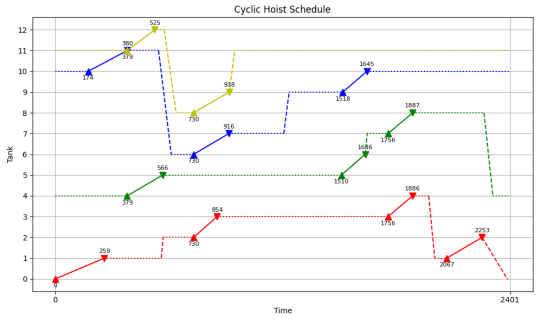
$$z^h_l \leq 2 - \left(x^k_{s,i} + y_{l,i}\right) \qquad \forall i \in N, \forall k \in \mathcal{K}, \forall l > i, \forall h < k \tag{19}$$

Constraint (18) ensures that if hoist k starts with move i and l is a smaller indexed move starting before i, then l cannot be done by a hoist h, which is greater than k.

These constraints, added to the model found in [3] were tested on many instances found in the literature. In Figure 4b you can see a solution, where the hoist trajectories are precisely determined.



(a) Original optimal schedule, only the loaded moves are determined



(b) Complete solution given by our model

Figure 4: The optimal solutions of a 4 hoists example, the hoists in increasing order have colors red, green, blue, yellow.

5 Summary

In this semester we reviewed the most important articles that formulated a MIP to solve the multi-hoist cyclic hoist scheduling problem. We gave a possible way of constructing the hoist trajectories, which - to our best knowledge - was missing from the literature. In the majority of the cases studied, the applied modifications did not change the optimal values. For visualization purposes a standardized process was created, which may be used in later projects.

References

[1] Phillips, L. W., and P. S. Unger. “Mathematical Programming Solution of a Hoist Scheduling Program.” A I I E Transactions 8 (2): 219–25. (1976).

[2] Emna Laajili. Modélisation et algorithmes pour le dimensionnement et l’ordonnancement cyclique d’atelier de traitement de surface. Automatique / Robotique. Université Bourgogne Franche-Comté, (2021).

[3] Leung, Janny MY, et al. "Optimal cyclic multi-hoist scheduling: A mixed integer programming approach." Operations Research 52.6 (2004): 965-976.

[4] Che, Ada, et al. "An improved mixed integer programming approach for multi-hoist cyclic scheduling problem." IEEE Transactions on Automation Science and Engineering 11.1 (2013): 302-309.

Table 2: Variables and Parameters for the Multi-Hoist Scheduling Problem

Symbol	Description
Variables	
z_i^k	$\begin{cases} 1 & \text{if move } i \text{ is done by hoist } k, \\ 0 & \text{otherwise,} \end{cases}$ for $i \in N, k \in \mathcal{K}$
$\mathscr{L}_i$	$\begin{cases} 1 & \text{if move } i \text{ is the last move for hoist 1,} \\ 0 & \text{otherwise,} \end{cases}$ for $i \in N^0$
y_{ij}	$\begin{cases} 1 & \text{if move } j \text{ starts after move } i, \\ 0 & \text{otherwise,} \end{cases}$ for $i, j \in N, i \neq j$
s_i	$\begin{cases} 1 & \text{if a panel is in process in tank } i \text{ at the beginning of the cycle,} \\ 0 & \text{otherwise,} \end{cases}$ for $i \in N$
t_i	start time of move i, for $i \in N^0$
C	hoist cycle time
Parameters	
$N, (N^0)$	number of moves, (move 0 included)
$\mathcal{K}$	set of hoists
d_i	the time to execute move i, for $i \in N^0$
$e_{i,j}$	$e_{i,j} = e_{j,i} =$ (empty) hoist travel time from tank i to tank j, for $i, j \in N$
L_i	the minimum soak time in tank i, for $i \in N$
U_i	the maximum soak time in tank i, for $i \in N$
M	an upper bound on the cycle time, e.g., $d_0 + \sum_{i=1}^{n}(d_i + L_i)$