Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 56-2 (2023) 5370–5375

Energy-oriented crane scheduling in steel coil storages: Evaluation of heuristic approaches

Patrick Oetjegerdes*. Patrick Schwier*. Christian Weckenborg*. Thomas S. Spengler*

*Institute of Automotive Management and Industrial Production, Technische Universität Braunschweig, Braunschweig 38106, Germany (Tel: +49-531-391-2210)
e-mail: {p.oetjegerdes, p.schwier, c.weckenborg, t.spengler}@ tu-braunschweig.de).

Abstract: Steel production is an energy-intense industry. Besides the production process itself, accompanying logistic processes are characterized by a high energy demand due to the weight of the products. We consider a steel coil storage, where a gantry crane stores and retrieves steel coils of up to 35 tons. An energy-oriented approach for the scheduling of crane operations can help save energy and lower operational costs. In industrial settings, crane operators require swift assistance in sequencing crane operations and assigning storage places for the coils. Therefore, we develop different heuristic approaches for the energy-oriented crane scheduling problem and compare them based on a case study derived from a German steel manufacturer.

Keywords: Job and activity scheduling, Production planning and control, Logistics in manufacturing, Sustainable Manufacturing, Operations research

1. INTRODUCTION

Cold rolled steel is an important product used in various industries, e.g., automotive manufacturing and white goods. Due to the high product variety regarding dimensions, weight, alloy type, and surface finish, it is a highly customized product manufactured make-to-order in a job shop production. Here, steel coil storages decouple the production process. *Incoming coils* must be stored to avoid blocking of preceding production stages (*storing order*), while *outgoing coils* must be retrieved to avoid starving of succeeding production stages (*retrieval order*). As coils are stored in a triangular stacking pattern, coils may have to be moved to access an outgoing coil. Accordingly, we refer to these as *reshuffled coils*.

Energy consumption of:

- Lifting and dropping a **15 t** coil: 0.69 kWh
- Lifting and dropping a **30 t** coil: 2.80 kWh

- Moving one meter in direction **a**: 0.14 kWh
- Moving one meter in direction **b**: 0.51 kWh

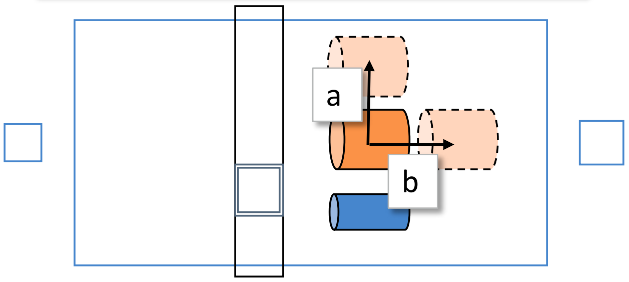

Fig. 1: Energy consumption of different crane and coil movements

Gantry cranes serve these storages. They consist of *portal*, *trolley*, and *hook*. Portal and trolley can move independently and with different velocities. A trolley movement consumes significantly less energy than a portal movement. The main driver of energy consumption for these movements is the traveled distance. If the hook lifts coils, however, the coil weight significantly affects the consumed energy as well (cf. Fig. 1) (Labitzke, 2011). Consequently, considering the energy consumption in an energy-oriented approach to crane scheduling may lead to significant energy savings. Therefore, we propose a new method to schedule the necessary movements of the crane and assign places to the steel coils to minimize the energy consumption of the crane.

The problem at hand consists of two interconnected planning problems: the scheduling of the crane movements to store, retrieve, and reshuffle steel coils, as well as to assign storage places to those in need of storage or reshuffling. In academic literature, literature associated with steel coil storage and container terminals considers these planning problems combined. For steel coil storage, Zäpfel and Wasner (2006) consider the scheduling and placement of incoming and outgoing steel coils to minimize the makespan of all orders. They propose a mixed integer non-linear program (MILNP) and solve the problem using a custom neighborhood search algorithm for large instances of up to 1,500 storage places and 500 orders. Yuan and Tang (2017) propose a MILNP to minimize the makespan of all orders. Using their time-space network flow model formulation, they can solve small instances optimally. They use an approximate dynamic programming algorithm to solve medium-sized instances with up to 76 storage places and 36 orders heuristically. However, neither of these approaches considers the energy consumption of cranes.

Further approaches address problems arising in container terminals. Galle et al. (2018) are the first to simultaneously

10.1016/j.ifacol.2023.10.183

consider place assignment and crane scheduling for yard cranes. They use a binary integer program to minimize the makespan of all orders, where the sequence of orders is fixed. Using a local search heuristic, they analyze the trade-off between current crane travel time and consideration of future reshuffles for an order. Heshmati et al. (2019) consider a combined place assignment and crane scheduling problem, with deadlines for outgoing and incoming coils and two cranes with non-interference constraints. Besides a mixed-integer linear programming approach to minimize the tardiness of operations and total storage costs, they develop a neighborhood search-based heuristic to solve large instances. However, neither of these approaches directly considers the energy consumption of cranes. Additionally, contributions to this literature stream neglect the triangular stacking pattern operated in steel coil storage.

The outcome of our literature overview is twofold: firstly, only a few papers consider storing and retrieval orders in crane scheduling approaches applied to steel coil storage. Secondly, in the general literature on crane scheduling, the energy consumption of cranes is not considered. Our main contributions are: For the first time, we define the energy-oriented crane scheduling problem and specify it for steel coil storage. Additionally, we develop and evaluate different variants of a two-stage tabu search-based heuristic to solve large instances based on the storage and production plan of a German steel manufacturer.

2. PROBLEM DESCRIPTION

Four main aspects characterize the problem at hand. They comprise the coils, the storage area, the gantry crane, and the objective. The coils $a \in A$ are customer-specific and therefore have a prespecified identity. Each coil has a specific weight m_a. The incoming coils $a \in A_{\text{in}}$ are placed on the input point and are associated with a storage order $\lambda \in \Lambda$ and its time window $\sigma_\lambda = [\sigma_\lambda^-, \sigma_\lambda^+]$. Outgoing coils $a \in A_{\text{out}}$ must be retrieved and transferred to the output point. Every outgoing coil is associated with a retrieval order $\lambda \in \Lambda$ and its time window $\omega_\lambda = [\omega_\lambda^-, \omega_\lambda^+]$. The deadline τ_λ for a storing order is equal to σ_λ^+, while for a retrieval order it is equal to ω_λ^+. All coils are part of the set $a \in A$, with $A_{\text{in}} \subseteq A$ and $A_{\text{out}} \subseteq A$. Coils can be both incoming and outgoing, $a \in A_{\text{in}} \cap A_{\text{out}}$.

Second, the storage area consists of multiple *rows*. A row consists of multiple *positions* along the row. Coils can be stored in up to two *layers* according to the triangular stacking pattern. The structure of the *storage area* results in a set of storage places $k \in \Psi$. Together with the input point I and output point O, the *storage* with its set of all places $k, q \in \Phi = \{I, O\} \cup \Psi$ is formed. Each place $k \in \Phi$ can be characterized by its row r_k, position p_k, and layer l_k. The distance between the middle of rows is d^1, between the center of the positions d^2, and between the middle of layers d^3. Due to the triangular stacking pattern, not all free places are available for storing. Also, an occupied place might be blocked, i.e., up to two reshuffle operations may be required to retrieve the blocked coil (cf. Fig. 2).

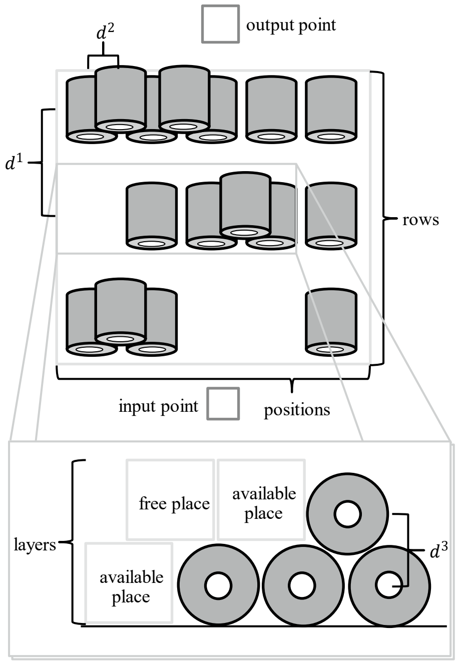

Fig. 2. Structure of a steel coil storage

Third, the elements of gantry cranes (portal, trolley, hook) require specification. Portal and trolley can move in different dimensions either simultaneously or successively. The hook can only move if neither portal nor trolley moves. Before portal or trolley movements may occur, coils must be lifted to the lifting height of the corresponding layer l_{H}. Different velocities and energy consumptions apply for each of the three types of movement. We define velocities v^1 for the portal movement, v^2 for the trolley movement, and v^3 for the hook movement. We assume that empty and loaded moves are performed in an alternating sequence. Consequently, the transport time for an empty move between two places (k, q) results to $t_{kq}^{\text{empty}} = \max\left(\frac{d^1 \cdot |r_k - r_q|}{v^1}; \frac{d^2 \cdot |p_k - p_q|}{v^2}\right) + \frac{d^3 \cdot (l_{\text{H}} - l_k)}{v^3} + \frac{d^3 \cdot (l_{\text{H}} - lq)}{v^3}$. Compared to an empty move, a loaded move requires additional time for closing and opening the hook for pickup and release of coils, which we refer to as μ. Therefore $t_{kq}^{\text{load}} = t_{kq}^{\text{empty}} + \mu$. We define the energy consumption of an empty move between the two places (k, q) as E_{kq}^{empty}. The energy consumption $E_{kq,a}^{\text{load}}$ of a loaded move additionally considers the weight of coils a.

Fourth, the objective for the crane scheduling problem requires justification. As time windows apply for each order and tardiness is not acceptable, time-oriented objectives are unsuitable. We identify the energy consumption as the only lever in economically and ecologically driving crane operations in steel coil storages and therefore aim to minimize the energy consumption of crane operations.

In conclusion, we decide which places get assigned to incoming and reshuffled coils as well as the exact time for each storing, retrieving, and reshuffling operation. We fulfill all storing and retrieval orders within the respective time windows, while minimizing the energy consumption resulting from the crane movements.

3. DEVELOPMENT OF THE HEURISTIC

The problem is complex to solve due to several reasons. A steel coil storage consists of a storage area with hundreds of places, of which 30–70 % are occupied with unique steel coils. During the day, up to 100 coils are moved. Due to being an operative planning problem, decisions about the scheduling of the crane operations must be made in a few minutes and multiple times a day. Finally, the problem at hand is a generalization of the classic job-shop scheduling problem, which is a well-studied NP-hard problem (Zäpfel & Wasner, 2006). It can also be reduced to a traveling salesman problem and is proven to be NP-hard in a strong sense (Tang et al., 2014). We, therefore, developed a two-stage heuristic approach to solve large real-life instances in reasonable time. It consists of two elements: an initial heuristic to generate a good and valid solution and a Tabu Search heuristic (TS) to improve the found solution. We propose and compare two approaches for the initial heuristic: First, a simple rule-based heuristic (RBH) derived from human experience. Second, an insertion-based heuristic (IBH) derived from pickup and delivery vehicle routing problems. For the TS we propose and compare two procedures to solve the problem of place assignment: First, a simple rule-based approach (PA-1). Second, a computationally more demanding approach evaluating all available places (PA-2).

A solution to our problem consists of a schedule $\Theta = (\lambda_1, \lambda_2, \ldots, \lambda_z)$ describing a sequence of orders $\lambda \in \Lambda$, conducted by the crane. z equals the number of incoming, outgoing, and reshuffled coils. Each order is associated with exactly one steel coil and a pickup and delivery place. We must assign a delivery place for incoming and reshuffled coils, while retrieval orders have a specified delivery place, i.e., the output point. A schedule is valid if all orders are fulfilled such that incoming coils are retrieved from the input point within their time window and outgoing coils are placed on the output point within their time window. The crane can, at any point, wait before picking up the next coil.

For our heuristic, we decompose our problem into two interdependent planning problems (sequencing and place assignment) for iterative consideration. We construct a sequence of orders, assign places to incoming and reshuffled coils, and afterward check if the resulting schedule is valid, considering the time for all necessary crane movements. If the resulting schedule is invalid, we consider a simple repair function.

Regarding the sequencing, it is important to consider the deadline τ_λ for each order. The deadline for storing orders is the point in time when an incoming coil is picked up from the input point at the latest. On the other hand, the deadline for retrieval orders is the point in time, at which a coil needs to be placed at the output point at the latest. In other words, the beginning of the loaded move is important for storing orders, while for retrieval orders, the end of the loaded move is. For our heuristic, we harmonize the deadlines for both order types to the point in time when the crane needs to pick up the coil. To accomplish that for retrieval orders, we need to consider the time necessary for the loaded move from the outgoing coil's current place k to the output point O, t_{kO}^{loaded}. Therefore, we modify the time window for placing the outgoing coil to a time window for picking up the outgoing coil $[\omega_\lambda^- - t_{kO}^{\text{loaded}} = \omega_\lambda^{\text{pickup}-}, \omega_\lambda^+ - t_{kO}^{\text{loaded}} = \omega_\lambda^{\text{pickup}+}]$ and the deadline to $\tau_\lambda = \omega_\lambda^{\text{pickup}+}$. However, the current place k of the outgoing coil at the time of retrieval cannot be determined ex-ante, as it may change over time. For the calculation, we assume that an outgoing coil remains in place until it is picked up. With this assumption, we can calculate the approximate time window for picking up outgoing coils.

3.1 Initial heuristics

The *rule-based heuristic* (*RBH)* is based on experiences from practice and follows four principles. Firstly, as trolley movements consume significantly less energy than portal movements, they may be favorable to the latter. Secondly, limiting crane movements to either a trolley or a portal movement effectively reduces the crane energy consumption, as the activation energy required to move either one is significant. Thirdly, reshuffle movements induce high energy consumption and should be avoided. Fourthly, a necessary reshuffle movement should be sequenced directly before the corresponding retrieval order, which may minimize crane movement due to the orders' spatial proximity.

For incoming and reshuffled coils, we need to assign a place in the storage area, identified by its layer, position, and row. For layer assignment, we differentiate between the types of coils to avoid reshuffles. Some coils are part of the set of outgoing coils. Outgoing coils $a \in A_{\text{out}}$ can be placed on top of other coils which are not part of the outgoing coils set, $a \notin A_{\text{out}}$. Coils $a \notin A_{\text{out}}$ are placed in the lowest not fully occupied layer. This leveling heuristic is adapted in practice, as it is effective in minimizing future reshuffles (Galle et al., 2018). In addition to layer assignment, we need to decide on row and position assignment. For incoming coils, we cannot avoid portal movement. Therefore, we try to avoid or minimize trolley movement by assigning a position as close to the axis of input and output point. If more than one place is available on this axis, we choose the row as close to the input point as possible. For reshuffled coils, the portal movement can be avoided if we assign a place in the same row. Therefore, we try to assign a position in the same row. Otherwise, we choose a position in a row with minimal portal movement. The change of rows should be directed toward the output point to reduce the portal movement required at the time of retrieval. Place assignment and order sequencing is conducted according to the following five steps in the rule-based heuristic.

Step 1: Create a list of orders Θ_n for this iteration n, consisting of the order $\lambda_{\min}$ with the earliest deadline τ_λ and all orders, whose time windows overlap with this order ($\omega_\lambda^{\text{pickup}-} < \tau_{\lambda_{\min}}$ or $\sigma_\lambda^- < \tau_{\lambda_{\min}}$).

Step 2: Add the order λ_i from the list Θ_n to Θ, whose pickup place is in the row the crane is currently positioned in to, avoid portal movements. If no order starts in the same row, add the order to the sequence, which starts in the closest row to the current crane position.

Step 3: If a reshuffle movement is necessary, sequence it right before the chosen order.

Step 4: Assign storage places for incoming and reshuffled coils. If $a \in A_{\text{in}} \setminus A_{\text{out}}$ assign the place in the lowest not fully occupied layer as close to the axis of input and output points. If $a \in A_{\text{in}} \cap A_{\text{out}}$, a place in any layer can be assigned. Again, the place should be located as close to the axis of input and output point. For reshuffled coils, assign a place in the same row. If no place is available within the same row, assign a position in a row closer to the output point with minimal portal movement. If no place is available in a row closer to the output point, assign a place in any row with minimal portal movement.

Step 5: Repeat *Step 1–4* until all orders are sequenced and places are assigned. If this leads to an invalid schedule, i.e., a time window is violated after the consideration of the time necessary for the crane movements, shift the corresponding order (and its associated reshuffles, if any) by one position earlier in the sequence and reassign places if necessary.

The *insertion-based heuristic* (*IBH)* uses that storing orders have a predetermined sequence as we cannot decide in which sequence the coils arrive at the input point. They must be stored before the next coil arrives. This reduces the sequencing problem to an insertion problem. It is favorable to sequence retrieval orders first since they affect the timing of necessary reshuffling orders and both pickup and delivery place are predetermined, which allows us to evaluate different insertion points for the storage orders in the first place. Like the rule-based heuristic, the necessary reshuffle operations are sequenced right before the associated retrieval order. The initial sequence of the retrieval orders is based on their deadlines. Since storing orders end at the assigned place in the storage, it is advantageous to sequence a storage order right before a retrieval order and assign a place ideally in the same row from which the subsequent order must be retrieved without blocking it. Therefore, portal movement is avoided. IBH consists of the following five steps.

Step 1: Create the sequence of all retrieval orders $\lambda \in \Lambda$ for outgoing coils $a \in A_{\text{out}} \setminus A_{\text{in}}$ based on their deadlines τ_λ.

Step 2: Check for necessary reshuffles and add them right before the corresponding retrieval order. Assign a place ideally in the same row as the reshuffle to avoid portal movement, otherwise assign a place in the same position as the retrieval order to avoid trolley movement. If neither is available, assign a place as close as possible without blocking any known outgoing coil.

Step 3: For all storing orders, add a storing order i before the retrieval order $i+1$ (and its associated reshuffles), if $\sigma_i^- < \omega_{i+1}^{\text{pickup}-}$. Assign a place for the incoming coil in the same row as the retrieval order. Otherwise, assign a place in the same position as the place of the next outgoing coil and avoid trolley movements. If neither is available, assign a place as close as possible to the input point without blocking another known outgoing coil.

Step 4: If necessary, add retrieval orders for outgoing coils $a \in A_{\text{in}} \cap A_{\text{out}}$. Insert them into the sequence at the latest possible position based on their deadline.

Step 5: Check if any time window gets violated. If a time window of any order gets violated, shift the corresponding order by one position earlier in the sequence and reassign places, if necessary.

3.2 Improving heuristic: Tabu Search

Based on the success of other researchers applying neighborhood search-based methods, we decided to use Tabu Search as our improving heuristic. The main components characterizing the TS are the neighborhood structure, the tabu structure, the tabu tenure, the diversification strategy, the aspiration criterion, and the stopping criterion. These parameters are presented in the following.

The *neighborhood structure* is explored by applying different operators to change the current solution. The neighborhood $N(\Theta_{\text{curr}})$ is generated by an insertion operator applying insertion moves. An insertion move (λ_i, j) represents selecting an order λ_i and inserting it at position j. $N(\Theta_{\text{curr}})$ comprises the insertion moves of every $\lambda_i \in \Theta_{\text{curr}}$ at every position in Θ. Thus, the number of solutions in $N(\Theta_{\text{curr}})$ is limited by the size of the neighborhood, called Z. The size of $N(\Theta_{\text{curr}})$ has a major influence on the runtime of a TS since each solution of $N(\Theta_{\text{curr}})$ must be evaluated by the objective function. In the problem at hand, however, some insertion moves of λ_i lead to invalid solutions by violating precedence or time window constraints. For example, inserting a reshuffle behind the corresponding retrieval order produces an invalid solution. For this reason, the number of possible insertion moves of λ_i can be limited considering precedence and time window constraints. Another factor influencing the feasibility of a neighboring solution is the assigned place in the case of a storage or reshuffling order. The availability of storage places depends on previous crane operations. As a result, the assigned place of an inserted order λ_i at position j may not be available at that time. To avoid this problem, assigned places of all storage and reshuffling orders are re-determined after the insertion move. Two different procedures are distinguished for place assignment, whose benefits we investigate within the case study. The first method for place assignment (PA-1) is the procedure already applied in step 2 of the IBH for reshuffled coils and step 3 of the IBH for incoming coils. The second method for place assignment (PA-2) calculates for each available place $q \in \Phi$ the sum of the energy consumption of the crane movements from the pickup place of λ_i to q and from q to the pickup place of λ_{i+1}. PA-2 assigns the place with minimal energy consumption.

The *tabu structure* applied in this TS is based on the insertion moves performed. For this purpose, a move (λ_i, j) is stored in combination with the duration of the tabu state. For a move (λ_i, j) performed in iteration k, the inverse move (λ_i, i) leading back to the previous solution is set tabu until iteration $k+\tau$. In case $j = i+1$ or $j = i-1$ an additional move $(\lambda_i + 1, i + 1) = k + \tau$ respectively $(\lambda_i - 1, i-1) = k+\tau$ must be added to the tabu list to avoid a reversion to the previous solution. The duration of the tabu state, called *tabu tenure*, has a major influence on the effectiveness of the TS. A distinction can be made between static and dynamic tabu tenures. The method applied in this paper uses a dynamic tabu tenure according to Glover and Laguna (1997) where each tabu move is individually assigned a random tabu tenure in the interval $[\tau_{\min}, \tau_{\max}]$. Since effective tabu tenures often depend on the size of the problem instance, $\tau_{\min} = 0.25 \cdot Z$ and $\tau_{\max} = 0.5 \cdot Z$ are used due to their successful application in similar problems (Landrieau et al., 2001).

Diversification strategies are intended to guide the search into unexplored regions of the solution space and are often based on a long-term memory. The strategy used in this approach is in literature referred to as frequency diversification. In this strategy, the frequency $f(\lambda_i, j)$ of an executed move *(λ_i, j)* is recorded and used as a penalty for evaluating this solution. The penalty costs are represented by the linear function $F(\delta, f) = \delta \cdot f(\lambda_i, j)$ with weighting factor δ and added to the evaluation of move *(λ_i, j). Aspiration criteria* are used to revoke the tabu state of a move. The most used criterion is called global aspiration. It revokes the tabu state of a move if this move yields a better solution than the best solution visited so far. Global aspiration is also used in our approach, as it guarantees that the solution obtained has not been visited previously. The designed TS employs a simple *stopping criterion* terminating the search process after a specified number of successive iterations k_{exit} with no improvement of Θ^*.

The performance of the TS heuristics depends on two parameters, namely, the diversification strategy δ and the stopping criterion k_{exit}. We evaluate different values for both parameters in a configuration run in two configuration scenarios (CS). Configuration scenario 1 (CS-1) with a small storage with 28 storage places, 7 storing, and 8 retrieval orders, and configuration scenario 2 (CS-2) with a large storage with 513 storage places, 15 storing, and 15 retrieval orders. Both CS have 50 % initial storage occupation and time windows of 20 minutes for outgoing coils. For both CS, we randomly generate 5 instances differing only in the initial place assignment of coils already in the storage. We use the IBH to generate the initial solution and PA-2 for assignment of new places. In both CS it can be observed that with an increase of k_{exit} to $k_{\text{exit}} = 100$ the quality of the solutions found also increases. A further increase up to $k_{\text{exit}} = 500$ yields no further improvement. Compared to $k_{\text{exit}} = 100$, $k_{\text{exit}} = 500$ has a higher solution time by a factor of 4 to 5 in both CS. Due to significantly lower solution times, k_{exit} is set to 100. In CS-1, a value of $\delta = 0.8$ reduces the average objective value by 0.13% compared to $\delta = 0.2$ in the case of $k_{\text{exit}} = 100$. With $k_{\text{exit}} = 500$, no improvement of the solution quality is achieved by a variation of δ. In the case of CS-2, there are no effects on the average objective value due to changes in δ between $k_{\text{exit}} = 100$ and $k_{\text{exit}} = 500$. Based on the lower objective value in CS-1, δ is fixed at 0.8.

4. CASE STUDY AND RESULTS

Our case study is based on a steel coil storage of a German steel manufacturer. It consists of 19 rows with 27 positions each, resulting in 513 storage places. At the start of an 8-hour shift, 15 storing orders with their respective time windows are submitted, as well as deadlines for 15 retrieval orders. Storing and retrieval orders are based on the production plans of the preceding and succeeding stages of the value chain. The distance between the middle of rows d^1 is 2.2 m, between the center of the positions d^2 0.8 m, and between the middle of layers d^3 1 m. The coils must be lifted to the lifting layer $l_{\text{H}} = 5$. This corresponds to a lifting height of 5 m. The velocity of the portal is 1.667 m/s, of the trolley 0.833 m/s, and of lifting and dropping 0.2 m/s. The hook needs 20 s to attach to a coil and 15 s to release it. The energy consumption of the crane movements in kWh between place k and q is the sum of the energy consumption of the portal, calculated by (1), of the trolley, calculated by (2), and in case of a loaded move the energy consumption of the lifting and dropping of the coil, calculated by (3). In case of empty moves between two places, the energy consumption of lifting and lowering the empty hook is assumed to be 0 kWh. The energy consumption functions for the crane movements are based on Labitzke (2011).

$$E^1_{kq} = \begin{cases} 0, \text{if } r_k - r_q = 0 \\ 0.5033 + 0.0041 \cdot |r_k - r_q| \cdot d^1, \text{else} \end{cases} \tag{1}$$

$$E^2_{kq} = \begin{cases} 0, \text{if } p_k - p_q = 0 \\ 0.1349 + 0.0014 \cdot |p_k - p_q| \cdot d^2, \text{else} \end{cases} \tag{2}$$

$$E^3_{kq,a} = (l_{\text{H}} - l_k) \cdot d^3 \cdot 0.01125 \cdot e^{0.1181 \cdot m_a} + (l_{\text{H}} - l_q) \cdot d^3 \cdot 0.02964 \cdot e^{0.0583 \cdot m_a} \tag{3}$$

We set up new scenarios to evaluate different effects of planning flexibility and operational settings. To evaluate the impact of increased flexibility, the retrieval orders have a fixed deadline τ_λ, but their time window spans 10, 20, or 30 minutes, respectively. We also study the influence of initial storage occupation levels by varying them between 30 %, 50 %, and 70 %. This results in 9 different scenarios. We generate 5 instances for each scenario, leading to 45 instances. The same coils are placed initially in the storage in each scenario, but their placement differs between instances. We first compare the two initial heuristics. Both were implemented with Python 3.10 and solved in less than 0.1 s with an Intel i7-1185G7 with 3.0 GHz and 32 GB RAM. The results are shown in Table 1.

Table 1. Objective values of the initial solution.

Scenario	Average objective value [kWh]		Difference [%]
	IBH	RBH	
30 %, 10 min	77.27	82.97	7.39
30 %, 20 min	77.20	83.35	7.96
30 %, 30 min	77.34	83.35	7.78
50 %, 10 min	77.41	83.49	7.86
50 %, 20 min	77.63	83.84	8.01
50 %, 30 min	77.77	83.76	7.70
70 %, 10 min	80.70	88.78	10.01
70 %, 20 min	80.78	88.85	9.99
70 %, 30 min	80.79	88.85	9.98

The IBH vastly outperforms the RBH. On average, the energy consumption is 8.52 % higher using RBH, ranging between 6.13 % and 12.60 %. It can also be stated that neither of the two heuristics can profit from the increased flexibility due to longer time windows. While the IBH outperforms the RBH regarding the objective value, the RBH has the advantage that the rules are easy to follow, even for a crane operator without computational support, and require little information about upcoming orders. This makes it a good starting point for an on-line approach. Nevertheless, since the IBH finds better solutions, we use it to generate the initial solution for the TS.

Next, we compare the two different approaches to assign places during the TS: PA-1 and PA-2. The results are shown in Table 2. The TS using PA-2 for assigning places outperforms the PA-1 approach. On average, the energy consumption is 1.6 % lower with PA-2. For low storage occupancy, the improvement is higher than for high occupancy. This can be explained by the fact that fewer places are available in a high occupancy setting, which often leads to PA-1 and PA-2 choosing the same place. Also, both TS procedures can profit from the increased flexibility due to longer time windows, even though the improvement is small. Additionally, we observe that the increased occupancy levels lead to higher energy consumption for the same orders in high occupancy settings. The average energy consumption between all instances with 50 % initial occupancy is 4.1 % higher for PA-1 and 5.3 % higher for PA-2, compared to the average of all instances with 70 % initial occupancy.

Table 2. Results of the Tabu Search Heuristic

	Average Objective Value [kWh]			Improvement [%]	
Scenario	IBH	PA-1	PA-2	PA-1	PA-2
30 %, 10 min	77.27	75.59	73.62	2.17	4.72
30 %, 20 min	77.20	75.53	73.52	2.17	4.77
30 %, 30 min	77.34	75.50	73.52	2.38	4.94
50 %, 10 min	77.41	75.49	74.12	2.47	4.24
50 %, 20 min	77.63	75.44	74.05	2.82	4.61
50 %, 30 min	77.77	75.14	74.05	3.38	4.79
70 %, 10 min	80.70	78.46	78.22	2.78	3.08
70 %, 20 min	80.78	78.41	77.86	2.93	3.62
70 %, 30 min	80.79	78.36	77.86	3.01	3.63

Table 3. Solution time of the Tabu Search approaches

	Average solution time [min]		Maximum solution time [min]	
Scenario	PA-1	PA-2	PA-1	PA-2
30 %, 10 min	0.31	4.21	0.41	5.03
30 %, 20 min	0.29	3.29	0.40	3.69
30 %, 30 min	0.34	3.99	0.51	4.84
50 %, 10 min	0.47	9.01	0.82	14.53
50 %, 20 min	0.41	5.57	0.66	7.88
50 %, 30 min	0.51	6.66	0.67	8.49
70 %, 10 min	0.66	14.11	1.00	26.31
70 %, 20 min	0.65	11.16	0.86	15.90
70 %, 30 min	0.68	12.01	0.93	16.31

The solution times of the different approaches to the TS are listed in Table 3. With solution times of up to 26.3 minutes for an instance with 70 % occupation and 10-minute time windows, the PA-2 approach, while proving better solutions, is taking too long for decision support in an operative setting. On the other hand, the PA-1 approach takes less than a minute to find a solution in every instance, which makes it suitable for the intended use.

5. CONCLUSION

We defined the energy-oriented crane scheduling problem in steel coil storage. To solve large instances, we developed two approaches to find an initial solution and evaluated them in a case study based on a real setting of a German steel manufacturer. Additionally, we developed a Tabu Search heuristic to improve the initially found solution. We developed two procedures to solve the place assignment problem occurring during the TS. While the TS with both place assignment procedures improved the initially found solution by 2.7 % and 4.3 %, respectively, the latter approach suffers from high computational times. Concluding, the approach of solving the problem with an insertion-based heuristic and a tabu-search with a rule-based place assignment can improve the crane scheduling in the steel coil storage from our partner from practice due to its fast solution time and high-quality solutions.

REFERENCES

Galle, V., Barnhart, C., & Jaillet, P. (2018). Yard Crane Scheduling for container storage, retrieval, and relocation. *European Journal of Operational Research*, *271*(1), 288–316.

Glover, F., & Laguna, M. (1997). *Tabu Search*. Springer US, Boston.

Heshmati, S., Toffolo, T. A., Vancroonenburg, W., & Vanden Berghe, G. (2019). Crane-operated warehouses: Integrating location assignment and crane scheduling. *Computers & Industrial Engineering*, *129*, 274–295.

Labitzke, N. (2011). *Wertorientierte Simulation zur taktischen Planung logistischer Prozesse der Stahlherstellung*. Gabler, Wiesbaden.

Landrieau, A., Mati, Y., & Binder, Z. (2001). A tabu search heuristic for the single vehicle pickup and delivery problem with time windows. *Journal of Intelligent Manufacturing*, *12*, 497–508.

Tang, L., Xie, X., & Liu, J. (2014). Crane scheduling in a warehouse storing steel coils. *IIE Transactions*, *46*(3), 267–282.

Yuan, Y., & Tang, L. (2017). Novel time-space network flow formulation and approximate dynamic programming approach for the crane scheduling in a coil warehouse. *European Journal of Operational Research*, *262*(2), 424–437.

Zäpfel, G., & Wasner, M. (2006). Warehouse sequencing in the steel supply chain as a generalized job shop model. *International Journal of Production Economics*, *104*(2), 482–501.