Adaptive Model and Threshold Algorithm for Hot Rolling Scheduling

S. I. Fainshtein, A. S. Fainshtein, V. E. Torchinsky, A. B. Belyavsky

Nosov Magnitogorsk State Technical University, Magnitogorsk, Russia

Abstract. Hot rolling batch scheduling problems are NP-hard and have a large number of multi-criteria constraints that do not allow to develop a feasible solution. The goal of this research is to generate plans with minor technological violations quickly and efficiently and to avoid any serious violations. A standardized method of transforming the problem with technological constraints into a constrained optimization problem and a heuristic threshold algorithm are proposed. The algorithm threshold system is determined by penalty constants. An equivalence relation is introduced for threshold systems. The threshold algorithm generates the same plan for any two equivalent threshold systems. An effective algorithm for automatic selection of penalty constants based on real data is also proposed. The model was tested at plate rolling shops of the Magnitogorsk Iron and Steel Works with the purpose of scheduling manufacture, storage and shipment of flat rolled products.

Keywords: hot rolling batch scheduling, dynamic scheduling, threshold algorithm, heuristics.

DOI 10.14357/20718632210310

Introduction

Issues encountered in the process of hot rolling scheduling represent very complex optimization problems with multiple constraints and a very large size of searching space. Technological constraints are ranked according to the "severity" level of violations. In general, a feasible solution for the actual daily order portfolio does not exist. Each solution involves certain violations of technological constraints. At the same time, any technological violation results in the rejection of commercial products in proportion to the "severity" level of violations.

This study is mainly devoted to the hot rolling scheduling problem. This problem is a well-known subject in production scheduling. The traditional approach to address such problem is to replace the constraints by penalty functions with subsequent convolution of individual partial criteria of the initial multi-criteria problem and then to minimize the generalized objective criterion.

Balas (1989) presented the prize collecting traveling salesman problem and distinguished two tasks: selection slabs (choosing slabs from an inventory (slab yard)) and sequencing (determining a sequence for processing the orders) [1]. Kosiba, Right, & Cobbs (1992) minimized the generalized objective function that reflected penalties for big jumps in width, gauge (thickness) and hardness [2]. The plan with the lowest penalty results in the lowest damage to the rolls and in the higher product quality. They described the scheduling problem as an asymmetric TSP where the objective was to minimize the total penalty. Cowling (1995) described a problem of generating a production plan for a steel hot rolling mill as a Prize Collection Vehicle Routing Problem (PCVRP) where each slab had a "prize" associated with rolling it and solved it by a heuristic method based on local search and Tabu Search (TS) [3]. Lopez, Carter, & Gendreau (1998) formulated the scheduling problem as a mathematical program and proposed a

heuristic method based on Tabu Search [4]. Tang, Liu, Rong, & Yang (2000) described the scheduling problem as a multiple traveling salesman problem (MTSP) and solved it using a modified genetic algorithm (MGA) [5]. Jia, Yi, Yang, Du, & Zhu (2013) described the problem as a multi-objective prize collecting vehicle routing problem (PCVRP) [6]. In order to avoid the selection of weight coefficients encountered in single objective optimization, they presented the multi-objective optimization algorithm based on Pareto-dominance.

Zixuan, Tieke, & Bailin (2016) proposed a batch scheduling model that takes the machine adjustment and the idle time as optimization objective. To solve the problem, a hybrid variable neighborhood search combined with simulated annealing is designed [7]. Pan, Wang, Zhou, & Chen (2017) considered both integrated scheduling problem of continuous casting (CC) and hot rolling process (HR) and proposed a novel Extremal Optimization algorithm which combines an exact mathematical model on CC stage and a heuristic dispatching algorithm on HR stage [8]. Zhang, Pan, Gao, Zhang, & Chen (2018) considered a hot rolling batch scheduling problem in compact strip production by hybrid variable neighborhood search algorithm. This problem aims to determine a sequence of the sheet strips in a predetermined number of rolling turns with the objective of minimizing average thickness change of adjacent sheet strips in all rolling turns [9]. Pan, Gao, & Wang (2019) proposed an improved hot-rolling scheduling heuristic for a multi-objective hot-rolling scheduling problem in the compact strip production [10]. Hu, Zheng, Gao, & Pardalos (2019) maximized the total length of all scheduled coils is to reduce roll wear by a heuristic method combining an improved Ant Colony Extended algorithm with local search procedures [11]. Hu, He, Song, & Feng (2020) minimized the total penalty incurred by the differences in width, thickness, and hardness among adjacent slabs, as well as the electricity cost of the rolling process by a heuristic method based on improved genetic algorithm [12].

Özgür, Uygun, & Hütt (2021) reviewed 90 articles for scheduling at hot rolling mills from 1989 to 2020 and classified them according to the details of the respective investigation [13]. According to this review there are two approaches for solving multi-objective optimization problems. The first approach is the transformation of the multi-objective optimization problem to a single-objective optimization problem, then using algorithms designed for single-objective optimization problem. The second approach is to use algorithms designed for multi-objective optimization problems. Among the reviewed articles, 12 used the multi-objective solution methods, while 64 used the single-objective methods. One common approach for transforming a multi-objective optimization problem to a single-objective optimization problem in the reviewed articles is the following: Every objective is multiplied with a weight and these weighted objectives are summed up. The difficulty of this approach is finding the proper weights. In almost all of the reviewed articles, the answer to this question is rather left to the readers.

In this research we used the first approach and proposed a new and efficient heuristic algorithm for automatic selection of "proper weights" (we call them penalty constants or adjusting constants). The goal of this research is to generate a new schedule from scratch with minor technological violations in real time mode and to avoid any serious violations. This problem has two special aspects. Firstly, the decision-maker ranks technological constraints according to the "severity" level of violations. Secondly, due to the large number of constraints, every partial criterion corresponds to an entire group of a particular type of constraints where penalties are charged in proportion to the "severity" level of violations. Thus, when replacing constraints with penalty functions in order to distinguish between solutions with a large number of "minor" violations and solutions involving "serious" technological violations it is not enough to control the distribution of penalty amounts between individual partial criteria. The contribution of each individual element to the penalty amount must be controlled.

An adaptive optimization model and a threshold algorithm are proposed to achieve the set goal. We assigned technological constraints that were ranked in increasing order according to the "severity" level of technological violations to the penalty vector $\{F_i\}$, $F_i \in \mathbf{R}^+$, $i = 1, \ldots, v$ where v was the number of technological constraints. The penalty vector $\{F_i\}$ functions as a system of adjusting constants.

The penalty vector $\{F_i\}$ produces a threshold system $B_v = \{0, B[1], ..., B[2^v - 1]\}$ for the threshold algorithm. The threshold system functions as a system of natural energy levels and allows to control the cost of placing each separate element in the solution. Then, an equivalence relation is introduced for B_v-sequences. The threshold algorithm generates the same plan for any two equivalent B_v-sequences. In addition, a polynomial algorithm for automatic selection of penalty constants based on real data is proposed.

The model was tested at plate rolling shops of the Magnitogorsk Iron and Steel Works. The application of this model allowed to solve problems arising during manufacture: Devyatov, S. Fainshtein, Belyavsky, & Torchinsky (2009) [14], S. Fainshtein (2010) [15]; storage and shipment of flat rolled products: S. Fainshtein, Tutarova, Kalitayev, Bukreyev, & Kolesnikov (2007) [16], Kaplan, Devyatov, S. Fainshtein, Tutarova, & Kalitayev (2009) [17], Devyatov, S. Fainshtein, Tutarova, & Kalitayev (2008) [18]. However, the need for manual selection of penalty constants still represented the main disadvantage of the algorithm. Later, an algorithm for automatic selection of penalty constants was proposed by S. Fainshtein (2013) [19] but, unfortunately, it required too much computation time for dynamic scheduling.

In this article, the adaptive model and the threshold algorithm are modified. A new and efficient heuristic algorithm for automatic selection of penalty constants is proposed. A comparison between the performance of the model and the existing manual system using real data is also provided.

1. Problem I with Technological Constraints

The main problem of hot rolling scheduling involves the ordering of the initial n-element array according to general and specific technical restrictions that function as constraints. We refer to such restrictions as *technological* constraints. One of the technological constraints is the limited length of an ordered sequence. Therefore, the original sequence is divided into m pairwise disjoint ordered sets, $1 \le m < n$.

In the context of rolling batch scheduling, m refers to the daily number of rolling mill startups. In the context of storing finished products, m refers to the number of formed piles. The number of piles is initially unknown but its maximum value is limited since the warehouse has a finite area. Therefore, it can be assumed that the initial set A is divided into m pairwise disjoint ordered sets some of which can happen to be empty.

***Problem* I**

An n-element set A is given, positive integer m, $1 \le m < n$; k positive integer characteristics, $k \ge 2$ for each $a \in A$. There is also a set of v technological constraints, $v \ge k$ such as the constraints on the sequence order of elements, sequence lengths and element placement within an unacceptable range.

Question: Is there any partition of set A into m pairwise disjoint ordered sets $A_1, ..., A_m$ such that all the constraints are satisfied?

Note. Unfortunately, the slab temperature was missing from our real data. Therefore, this model provides no constraints related to the temperature of adjacent slabs. The specific type of constraints depends on particular technical instructions. Fainshtein (2010) and Devyatov et al. (2009) described the types of technological constraints involved with the problem of hot rolling batch scheduling in detail; Kaplan et al. (2009), Devyatov et al. (2008) and S. Fainshtein et al. (2007) described the types of constraints for the problem of storing and shipping finished flat rolled products.

2. Substituting Constraints with Cost Functions

We assign technological constraints existing in the Problem I, which are ordered based on the "severity" level of technological violations, to the penalty vector $\{F_i\}$, $F_i \in \mathbf{R}^+$, $i = 1, \ldots, v$ arranged in increasing order.

For example, the penalty vector for hot rolling mill 2000 of the Magnitogorsk Iron and Steel Works had the following structure: F_1 – penalty for placing an order within an unacceptable range; F_2 – penalty for exceeding the length of continuously rolled orders with the same width (or thickness); F_3 – penalty for exceeding the maximum batch size; F_4 – penalty for jump in hardness; F_5 – penalty for jump in thickness; F_6 – penalty for jump in width.

We refer to any splitting of the initial set A into m pairwise disjoint ordered sets as the partition U

of the Problem I. Then we assign the cost $f(U)$ to the partition U of the Problem I:

$$f(U) = \sum_{i=1}^{v} S_i F_i, \quad (1)$$

where S_i is the number of violations of the i-th constraint within the partition U. Thus, for the entire set of partitions U of the Problem I the cost function f is defined, which represents the cost of penalties for committed violations. The number of violations for each constraint $\{S_1(F_1,\dots F_v), \dots, S_v(F_1,\dots F_v)\}$ represents partial criteria.

We keep the j-th sequence, $j = 1, \dots, m$ unchanged in the partition U and consider two adjacent elements a_i and a_{i+1} within this sequence, where $1 \le i \le d(j) - 1$ and $d(j)$ is the length of the j-th sequence. The cost of changeover from a_i to a_{i+1} $Cost(a_i, a_{i+1}, j)$ is comprised of the amounts of penalties charged for the violations of constraints on the sequence order and the amount of penalties charged separately with a_i, and a_{i+1} for the violation of constraints on the sequence length and element placement within an unacceptable range.

Similarly, the value $Cost(a, j, r)$ of placing element a in r-th position of j-th sequence of length d, $r \le d(j) + 1$ is calculated. When the element a is placed in the r-th position of j-th sequence in U, the cost of placing the element a in partition U is

$$Cost(a, U) = Cost(a, j, r) \quad (2)$$

3. Threshold System

We assign the ordered penalty vector $\{F_i\}$, $i = 1, \dots, v$ to an ordered sequence $B_v = \{0, B[1], \dots, B[2^v - 1]\}$ where $B[j]$, $j = 1, \dots, 2^v - 1$ are the sums of different $\{F_i\}$ coordinates that are taken in sequence in the quantity of $C_v^1, \dots, C_v^v$ where C_v^i represents a binomial coefficient (the number of these sums is $2^v - 1$ since this is the sum of binomial coefficients). We refer to this sequence as a *complete* B-sequence.

For example, when $v = 3$, the complete B-sequence can be represented as $B_3 = \{0, a, b, c, a + b, a + c, b + c, a + b + c\}$ for the penalty system $\{F_1 = a, F_2 = b, F_3 = c\}$.

We arrange the complete B-sequence in lexicographical order and name it *a complete lexicographical B-sequence.* When $v = 3$, the complete lexicographical sequence is $B_3 = \{0, a, b, b + a, c, c + a, c + b, c + a + b\}$.

It is obvious that $\{F_i\}$, $F_i \in \mathbf{R}^+$, $i = 1, \dots, v$, can always be chosen in such a way that:

a) elements of the complete B-sequence are all different;

b) the complete B-sequence in increasing order matches with the complete lexicographical B-sequence.

We refer to a complete B-sequence as *regular* when it satisfies (a)-(b).

For example, when $v = 3$, the penalty vector $\{a = 1, b = 1.5, c = 2\}$ forms a non-regular complete B-sequence since $a + b = 2.5 > c = 2$. We increase the value of c. The penalty vector $\{1, 1.5, 3\}$ forms the following regular B-sequence: $\{0, a = 1, b = 1.5, a + b = 2.5, c = 3, c + a = 4, b + c = 4.5, c + b + a = 5.5\}$.

The regular B-sequence will be used as a threshold system of the algorithm and function as a natural system of energy levels.

Now we define a discrete function $p(t)$, $t = 0, 1, \dots 2^v - 1$, for which

$$p(t) = B[t] \quad (3)$$

The parameter t has the meaning of iteration index of the threshold algorithm; the value $p(t)$ (threshold value) sets a limit on the cost of placing an element in the current iteration. For example, when $v = 3$ and the threshold value $p(t)$ equals $a + b$ for the current iteration, an element can be placed with violations of the first and/or the second constraint whereas more "serious" violations of the third constraint are forbidden at this iteration.

4. Threshold Algorithm

Initially, all the elements of set A are not placed. We create the partition for Problem I (i.e. form m ordered sequences) gradually by starting with empty sequences. Specific cost formulas for placing individual elements and ordered sequences depend on particular technological instructions.

Initialization. We form a vector of positive real penalties that corresponds to the constraints of the problem and is arranged in a strictly ascending order. We produce a regular B-sequence (algorithm threshold system) based on the penalty vector.

Step 1. We set $t = 0$, $p(t) = B[t]$ (threshold cost of placing an element). We set minimum and maximum cut values as $MinCut = B[t]$ and $MaxCut = B[x_0]$, respectively.

Note. The x_0 value is selected based on specific technological constraints. *MaxCut* value must be lower than the value of any penalty corresponding to a serious technological violation.

Step 2. We rank all the unplaced elements according to any of k integer positive characteristics of the elements. Ranking characteristic and direction are selected depending on the content of specific technological instructions.

When $MinCut \geq MaxCut$, we move on to Step 4. Otherwise, when the cost $Cost(a_i, a_{i+1}, j)$ of changeover from a_i to $a_{i+1} > MinCut$, we cut the sequence between elements a_i and a_{i+1}, $i = 1, ..., d - 1$ where d refers to the length of the sequence of unplaced elements.

Step 3. We place the columns obtained at Step 2. We place a column in the j-th sequence, $j = 1, ..., m$ when the cost of its placement is less than $p(t)$. We repeat Step 3 for as long as there are any remaining placement costs that are $\leq p(t)$. If there are any unplaced columns remaining we move on to Step 4 or, otherwise, *Stop*.

Step 4. We split unplaced columns into separate elements and place an element in the j-th sequence, $j = 1, ..., m$ when the cost of its placement is $\leq p(t)$. We repeat Step 4 for as long as there are any remaining placement costs that are $\leq p(t)$. If there are any unplaced elements remaining we move on to Step 5 or, otherwise, *Stop*.

Step 5. $t = t + 1$. When $t \leq 2^v - 1$, $MinCut = B[t]$; $p(t) = B[t]$ and we return to Step 2 or, otherwise, *Stop*.

Note 1. The upper threshold $p(2^v - 1)$ is equal to the sum of penalties for all kinds of technological violations, which guarantees the placement of all elements.

Note 2. The penalty vector $\{F_i\}$ represents a vector of adjusting constants. By changing their values, the number of violations $\{S_1(F_1,\ldots, F_v), \ldots, S_v(F_1,\ldots, F_v)\}$ for each type of constraints can be changed.

Computational complexity. The computational complexity of the algorithm is $O(2^v n^3)$, v is the fixed number of technological constraints and n is the number of placed elements.

5. Constrained Optimization Problem II

Now we transform the Problem I with constraints into the Constrained Optimization Problem II, which allows us to generate solutions with a certain number of violations.

We assume that U refers to a partition of the Problem I; $u(t)$ refers to the current partition of Problem I formed by the threshold algorithm at the t-th iteration; $f(u)$, $u \in U$ refers to the cost of u calculated according to (1); $Cost(a, u(t))$ refers to the cost of placing a in $u(t)$ calculated according to (2); $p(t)$ refers to the threshold cost of placing a in the t-th iteration calculated according to (3).

***Constrained Optimization Problem* II**

Solve the problem to minimize $f(u)$, $u \in U$, subject to the constraints $Cost(a, u(t)) \leq p(t)$ for any $a \in A$, $t = 0, 1, ..., 2^v - 1$ where v is the number of technological constraints.

Thus, using the algorithm thresholds we control the number of possible violations for each placed element.

6. Automatic Selection of Adjusting Constants of the Threshold Algorithm

We denote subset $\mathbf{R}^v$ with positive coordinates as D. We assign each point included in D to the penalty vector $\{F_i\}$, $F_i \in \mathbf{R}^+$, $i = 1, \ldots, v$ arranged in non-decreasing order and to the complete B-sequence B_v.

Definition 1. Two complete B-sequences X and Y are referred to as isomorphic $X \sim Y$ when the following takes place for any k indices $1 \leq i(1) < ... < i(k) < n$ and m indices $1 \leq j(1) < ... < j(m) < n$:

$$X_{i(1)} + ... + X_{i(k)} < (\text{or} >, \text{or} =)\ X_{j(1)} + ... + X_{j(m)} \Leftrightarrow$$
$$Y_{i(1)} + ... + Y_{i(k)} < (\text{or} >, \text{or} =)\ Y_{j(1)} + ... + Y_{j(m)}.$$

Definition 2. When any two points taken from D form isomorphic sequences, they are referred to as equivalent.

The relationship defined in this way provides a partition of D into equivalence classes. It can be easily observed that the following statements are true:

Statement 1. The number of equivalence classes is finite.

Statement 2. D is a disjoint union of convex equivalence classes (classes are described as a finite system of linear inequalities).

Statement 3. For any two equivalent sequences the Threshold algorithm generates the same plan (violations of constraints allowed at the current iteration of the algorithm depend only on the sequence order of $B[i]$, $i = 1, ..., 2^v - 1$ and not on their values).

It is worth reminding that in regular *B*-sequences all elements are different and the ascending order matches with lexicographic order. To generate new plans without changing the order of the penalty vector components, lexicographic order of the *B*-sequence elements must be inverted. The given ranking of criteria allows to apply the idea of the successive concession method developed by Wentzel (1983) [20]. Our goal is to reduce the number of violations for the *i*-th partial criterion S_i without increasing the number of violations for criteria with greater importance $\{S_{i+1}, \ldots, S_v\}$.

For example, when $v = 4$ for penalty vector $\{a, b, c, d\}$, $c > a + b$ in a regular *B*-sequence. This means that the Threshold algorithm first places elements with technological violation of the first and/or the second type and only then it places the elements with violation of the third type. When we want to reduce the number of violations of the fourth type, there is no point to increase the penalty value *d* since it is the highest one already. By decreasing *c* to the value $b \leq c \leq a + b$ we change the order of element placement and obtain a new plan with more violations of the third type but less violations of the fourth type. Certainly, this is not always possible.

Thus, we obtain a heuristic algorithm for automatic selection of values for a single adjusting constant. When the number of violations S_i (F_1,… F_v) for the *i*-th partial criterion is too high in the resulting plan, the value of the previous penalty F_{i-1} should be shifted from the initial value of $F_{i-1} > \sum_{j=1}^{i-2} F_j$ to the value of $F_{i-2} < F_{i-1} < F_1 + F_2, i \geq 3$. Then the plan with minimum S_i value should be chosen from the resulting plans provided that the values of criteria with greater importance $\{S_{i+1}, \ldots, S_v\}$ have not decreased.

Automatic selection of a single adjusting constant value takes no longer than $2^{v-1} - v$ runs of the threshold algorithm where *v* is the number of technological constraints. During dynamic scheduling the F_{i-1} values can be changed incrementally to reduce the number of runs.

7. Practical Application of the Model

We consider one of the practical applications of the proposed model, which is the hot rolling batch scheduling applied at the hot rolling mill 2000 of the Magnitogorsk Iron and Steel Works [14]-[15]. Informally, this problem can be stated in the following way. The maximum number of 7 ordered sequences (batches) – a daily working plan of the mill – must be generated from the order portfolio for the current day (about 250-300 slabs). It is worth noting that the slabs have already been selected from an inventory and the daily order portfolio has already been generated.

Every order has a certain set of attributes such as width, gauge (thickness), rolled strip length, rolling group (hardness), application and so on. The most important constraints include change in width, gauge (thickness) and hardness (rolling group). There are six rolling groups and the allowed transitions between them are described in operating procedures. For example, only groups 5 or 6 are allowed to be rolled after groups 1 and 2. Other constraints such as the total rolling length, the length of continuously rolled orders with the same width (thickness) and placing an order within an unacceptable range also affect product quality.

The production is scheduled by a highly qualified operator. The analysis of daily plans prepared by the operator showed that the real plans always included a certain number of violations. We can say that the operator is guided by the following heuristic rule when scheduling the rolling sequence: “rolling from wide to narrow and from thin to thick”. Also, there are limits on kilometric ranges for orders included in the production batch, depending on their parameters or application. In addition, there are limits on the length of continuously rolled slabs with the same attributes where orders with the same width or thickness cannot be rolled for too long.

We had plans prepared by the operator for a daily order portfolio (7 production batches) at our disposal. We preliminarily removed the starting sections (the warming-up section in which the slabs are arranged from narrow to wide to warm up the rolls) from these production batches, then combined all the orders into a single sequence, randomly shuffled them and ran the Threshold algorithm 8 times (with different settings). The algorithm execution during each of the 8 runs took about one second.

The analysis of the plans (Tables 1 and 2) showed that the application of the software allowed to decrease the number of most serious technological violations (0 vs. 3), decrease the total number of all violations and increase the production batch length without rolling turns

Table 1. Plan comparison based on the number of violations

Number of violations	*Operator*	*Software*
Width	3	0
Thickness	16	17
Rolling group	17	8
Total:	36	25

Table 2. Distribution of production batch lengths

Production batch	1	2	3	4	5	6	7
Operator (km)	46	96	95	96	101	96	105
Software (km)	168	150	112	70	123	5	6

It is also interesting to compare the plans generated by the operator and the algorithm on a daily order portfolio provided that the number of the production batch where a slab must be rolled is known for each order. This process is called “hot charging” [21]: if a slab can be sent to the hot rolling mill without going to the slab yard, a lot of heating energy could be saved. We shall consider the example below. The penalty constants used by the algorithm are given in Table 3. The highest penalties (200 and 80) are imposed for violations associated with width.

Analysis of the number of violations committed by the operator and the software is shown in Table 4. The software managed to significantly decrease the number of violations related to width (1 vs 5) due to a certain increase in the number of violations related to thickness and rolling group.

Analysis of the production batches showed that the first three batches were the same in both plans but there were some differences in the fourth installation batch. Starting sections of the fourth production batch generated by the operator and the software are presented in Tables 5 and 6. The "Changeover cost" column in the i-th row shows the cost of changeover from (i-1)-th order to i-th one. This cost is composed of penalties charged for the following three main types of violations: width, thickness and rolling group. The operator committed the following three width-related violations: during changeovers from item 3 to item 4, item 4 to item 5 and item 5 to item 6 as well as one rolling group change during the changeover from item 4 to item 5.

Table 3. Penalty constants of the algorithm

Type of violation	*Penalty*
From narrow to wide	200
From wide to narrow at intervals of > 250 mm	80
Jump in rolling group	40
From thick to thin	25
The length of continuously rolled slabs with the same width or thickness is too high	20
Unacceptable range	10

Table 4. Plan comparison based on the number of violations

Number of violations	*Operator*	*Software*
Width (any)	5	1
Jump in rolling group	7	8
Jump in thickness	0	2
Unacceptable Range	0	0
Total:	12	11

Table 5. Starting section of the fourth production batch (Operator)

Item	*Order*	*Thickness (mm)*	*Width (mm)*	*Rolling group*	*Changeover cost*
1	71	6.00	1040	2	0
2	72	7.00	1040	2	0
3	73	6.00	1040	2	0
4	74	4.70	1060	2	200
5	75	3.00	1248	1	225
6	76	2.50	1311	1	200
7	77	2.00	1310	1	0
8	78	2.00	1310	1	0
9	79	2.60	1243	1	0
10	80	2.60	1243	1	0
11	81	2.60	1233	1	0
12	82	2.00	1213	1	0
13	83	1.80	1060	1	0
14	84	1.80	1060	1	0
15	85	1.50	1000	1	0
Total					**625**

Table 6. Starting section of the fourth production batch (Software)

Item	*Order*	*Thickness* (*mm*)	*Width* (*mm*)	*Rolling group*	*Changeover cost*
1	76	2.50	1311	1	0
2	77	2.00	1310	1	0
3	78	2.00	1310	1	0
4	75	3.00	1248	1	200
5	79	2.60	1243	1	225
6	80	2.60	1243	1	200
7	81	2.60	1233	1	0
8	82	2.00	1213	1	0
9	83	1.80	1060	1	0
10	84	1.80	1060	1	0
11	74	4.70	1060	2	25
12	71	6.00	1040	2	0
13	72	1.50	1040	2	0
14	73	1.50	1040	2	0
15	85	1.50	1000	1	25
Total					**50**

The software generated the following two rolling group changes: during changeovers from item 11 to item 12 and item 14 to item 15. The total penalty cost is 625 and 50 for the first and the second plans, respectively.

Conclusion

An adaptive optimization model and an effective threshold algorithm for addressing problems involved with hot rolling scheduling are created. The proposed model has the following advantages:

- a standardized method of transforming the problem with technological constraints into a constrained optimization problem allows to generate plans with minor technological violations;
- controlling the cost of placing each separate element in the solution allows to avoid serious technological violations;
- a flexible system of penalty constants allows to adapt the algorithm to real data;
- automatic selection of penalty constants based on real data;
- generating solutions in real time mode.

References

1. Balas, E. 1989. The prize collecting traveling salesman problem. Networks. 19:621–636. doi: 10.1002/net.3230190602.
2. Kosiba E. D., Right J.R., & Cobbs A. E. 1992. Discrete event sequencing as a traveling salesman problem. Computers in Industry. 19:317–327. doi: 10.1016/0166-3615(92)90069-Y.
3. Cowling, P. I. 1995. Optimization in Industry (ch. Optimization in steel hot rolling). Wiley, Chichester, England.
4. Lopez L, Carter M. W., & Gendreau M. 1998. The hot strip mill production scheduling problem: A tabu search approach. European Journal of Operation Research. 106(2-3):317–335. doi: 10.1016/S0377-2217(97)00277-4.
5. Tang L., Liu J., Rong A., & Yang Z. 2000. A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron & Steel Complex. European Journal of Operation Research. 124(2):267–282. doi: 10.1007/s10878-017-0130-4.
6. Jia S.J., Yi J., Yang G.K., Du B., & Zhu J. 2013. Multi-objective optimization algorithm for the hot rolling batch scheduling problem. International Journal of Production Research. 51(3):667-681. doi: 10.1080/00207543.2011.654138.
7. Zixuan W., Tieke L., & Bailin W. 2016. Hybrid variable neighborhood search for batch scheduling
1. of hot rolled steel tube. 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC): 335–338. doi:10.1109/IMCEC.2016.7867228.
8. Pan Z., Wang T., Zhou X., & Chen P. 2017. Application of extremal optimization approach to the
2. integrated scheduling problem of continuous casting and hot rolling process. 29th Chinese Control and Decision Conference (CCDC). 2017: 2429–2434. doi:10.1109/CCDC.2017.7978922.
8. Zhang B., Pan Q., Gao L., Zhang X., & Chen Q. 2018. A hybrid variable neighborhood
3. search algorithm for the hot rolling batch scheduling problem in compact strip production. Computers & Industrial Engineering. 116 (2018): 22 – 36. doi:10.1016/j.cie.2017.12.013.
9. Pan Q.-K., Gao, L., & Wang, L. 2019. A multi-objective hot-rolling scheduling problem in the compact strip production. Applied Mathematical Modelling. 73:327 – 348. doi:10.1016/j.apm.2019.04.006.
10. Hu W., Zheng Z., Gao X., & Pardalos P.M. 2019. An improved method for the hot strip mill
4. production scheduling problem. International Journal of Production Research. 57 (10):3238–3254. doi:10.1080/00207543.2019.1579932.
11. Hu Z., He D., Song W., & Feng K. 2020. Model and Algorithm for Planning Hot-Rolled Batch
5. Processing under Time-of-Use Electricity Pricing. Processes 8 (1): 42. doi:10.3390/pr8010042.
12. Özgür A., Uygun Y. & Hütt M.-T. 2021. A review of planning and scheduling methods for hot rolling mills in steel production. Computers & Industrial Engineering. 151:106606. doi: 10.1016/j.cie.2020.106606.

13. Devyatov, D. Kh., Fainshtein, S.I., Belyavsky, A.B., & Torchinsky, V.E. 2009. Evristicheskaya optimizatsionnaya model' dlya zadach s ogranicheniyami, voznikayushchikh pri operativnom planirovanii prokatnogo proizvodstva [A heuristic optimization model for constrained problems arising during dynamic scheduling of plate rolling]. Informacionnye tehnologii [Information technologies]. 9:16-20.
14. Fainshtein S.I. 2010. Adaptivnaya optimizatsionnaya model' dlya operativnogo planirovaniya zadach listoprokatnogo proizvodstva [Adaptive optimization model for dynamic scheduling of plate rolling problems existing in flat rolled product manufacturing]. Informatsionnye tekhnologii i vychislitel'nyye sistemy [Information technologies and computing systems]. 1:92-98.
15. Fainshtein S.I., Tutarova V.D., Kalitayev A.N., Bukreyev A.Y., & Kolesnikov Y.F. 2007. Operativnoye planirovaniye dvizheniya gotovoy produktsii na skladakh metallurgicheskikh predpriyatiy [Dynamic Scheduling of Finished Product Movement at Warehouses of Metallurgical Enterprises]. Vestnik Magnitogorskogo gosudarstvennogo tekhnicheskogo universiteta im. G.I. Nosova [Herald of the Magnitogorsk State Technical University n.a. Nosov]. 4(20):108-112.
16. Kaplan D.S., Devyatov D.Kh., Fainshtein S.I., Tutarova V.D., & Kalitayev A.N. 2009. Evristicheskiy polinomial'nyy algoritm operativnogo planirovaniya razmeshcheniya gotovoy produktsii na skladakh metallurgicheskikh predpriyatiy [Heuristic polynomial algorithm for dynamic scheduling of finished product allocation at warehouses of metallurgical enterprises]. Avtomatizatsiya. Sovremennyye tekhnologii [Automation. Modern technologies]. 6:35-39.
17. Devyatov D.Kh., Fainshtein S.I., Tutarova V.D., & Kalitaev A.N. 2008. Operativnoe planirovanie otgruzki gotovoj produktsii so skladov metallurgicheskikh predprijatij [Dynamic scheduling of finished product shipment from warehouses of metallurgical enterprises]. Mekhatronika, avtomatizatsija, upravlenie [Mechatronics, automation, control]. 4:36-40.
18. Fainshtein S.I. 2013. Adaptivnaya model' dlya zadach uporyadochennogo razbiyeniya s ogranicheniyami [Adaptive model for constrained problems involved with ordered partition]. Matematicheskoye i programmnoye obespecheniye sistem v promyshlennoy i sotsial'noy sferakh [Mathematical and software systems in industrial and social areas]. 1(3):53-56.
19. Wentzel E.S. 1983. Operations Research: a Methodological Approach. Moscow: Mir Publishers. 264 p.
20. Tang L., Liu J., Rong A., & Yang Z. 2001. A review of planning and scheduling systems and methods for integrated steel production. European Journal of Operation Research. 133(1):1-20.

Fainshtein S. I. Assistant professor of G.I. Nosov Magnitogorsk State Technical University, Magnitogorsk, 38 Lenina Ave. Associate professor. 11 published articles. Topics of interest: discrete optimization, information technologies. Corresponding author, e-mail: sfainshtein@yandex.ru

Fainshtein A. S. Assistant professor of G.I. Nosov Magnitogorsk State Technical University, Magnitogorsk, 38 Lenina Ave. PhD, associate professor. 40 published articles. Topics of interest: operators on Banach spaces, Lie algebras of operators, discrete mathematics. E-mail: swetlana@mgn.ru

Torchinsky V. E. Assistant professor of G.I. Nosov Magnitogorsk State Technical University, Magnitogorsk, 38 Lenina Ave. Associate professor, leading software engineer. 10 published articles. Topics of interest: optimization and evolution algorithms, information technologies. E-mail: vet@magtu.ru

Belyavskiy A. B. Assistant professor of G.I. Nosov Magnitogorsk State Technical University, Magnitogorsk, 38 Lenina Ave. Associate professor, leading software engineer. 15 published articles. Topics of interest: database, optimization, information technologies. E-mail: a.belyavsky@magtu.ru