Population-based Extremal Optimization Algorithm for Hot Rolling Scheduling Problem

Kai SUN 1,†, Genke YANG 2, Changchun PAN 2

1 School of Electronic Information and Control Engineering, Shandong Institute of Light Industry, Shandong, China
2 Department of Automation, Shanghai Jiao Tong University, Shanghai, China

Abstract

The paper presents a valid model and a solution method named population-based extremal optimization algorithm (PEOA) to solve the hot rolling scheduling problem (HRSP). Firstly, the problem is formulated as a prize-collecting vehicle problem (PCVRP), which considers two major requirements: (a) selecting a subset of slabs from manufacturing slabs to be processed; (b) determining the optimal production sequence under multiple constraints, such as sequence-dependant transition costs, non-execution penalties etc. Secondly, a new algorithm which combined the extremal optimization (EO) with population evolutionary technique is proposed to solve the problem. The proposed algorithm is applied to a set of real production data and the performance of the algorithm is evaluated by comparing the results with other algorithms such as EO, modified algorithm (MGA), etc. Comparing results indicate that this new algorithm is an effective and competitive approach for the HRSP.

Keywords: Hot Rolling Scheduling Problem; Extremal Optimization; Population Evolutionary Algorithm;

1. Introduction

Iron & steel industry is the cornerstone of manufacturing industries, providing the indispensable materials for other industries [1]. As one of the most important production processes in the steel industry, hot rolling connects upstream processes like continuous casting and downstream processes such as cold rolling, annealing, etc. The hot rolling scheduling problem (HRSP) consists of the generation of production schedules for the hot rolling phase. The HRSP is a very hard optimization problem which has several conflicting objectives and multiple constraints, and the solution to the problem should balance those conflicting objectives and constraints.

Hot rolling production scheduling is a NP-hard problem [1][2] and many researchers had studied in recent years. Lopez et al. [2] described hot rolling and its related production processes in detail. They formulated the scheduling problem as a mathematical program and proposed a heuristic algorithm based on tabu search (TS) to solve the problem. Chen et al. [3] modeled the batch rolling planning as a vehicle routing problem with time windows (VRPTW) where position limitation of some slabs in a round is considered. Tang et al. [4] formulated the problem as a multiple traveling salesman problem (MTSP) and proposed a modified genetic algorithm (MGA), the computational results show the method is very effective. Cowling et al. [5] presented a multi-agent architecture for dynamic scheduling of steel hot rolling problem. Tang & Wang [6] modeled the HRSP as a prize-collecting vehicle routing problem (PCVRP) and presents an iterated local search algorithm based on very large-scale neighborhood for solving the problem. Zhao et al. [7] formulated the problem as a vehicle routing problem with time windows (VRPTW) and proposed a
two-stage scheduling method based on a modified partheno genetic algorithm (PGA) associated with heuristic rules. Pan & Yang [8] presented a transparent model and a solution approach to solve a large-scale rolling batch scheduling problem and demonstrate the efficiency of the proposed algorithm.

In the paper, we formulated the production scheduling problem of the hot rolling as a prize-collecting vehicle routing problem (PCVRP), which considers two major requirements: (a) selecting a subset of slabs from manufacturing slabs to be processed; (b) determining the optimal production sequence under multiple constraints, such as sequence-dependant transition costs, non-execution penalties etc. And then, a new algorithm which combined the extremal optimization (EO) with population evolutionary algorithm is proposed to solve the HRSP. The proposed algorithm is applied to the real hot rolling production data to test its validity and efficiency.

The rest of the paper is organized as follows. In Section 2, we present the problem formulation for the HRSP. Section 3 gives the description of population-based extremal optimization algorithm (PEOA) for the HRSP. In Section 4, we provide experimental results and performance analysis by simulating production scale data of HRSP. Finally, Section 5 presents some concluding remarks of the paper.

2. Problem Formulation

2.1. Constraints

The hot rolling scheduling is to generate a reasonable mill sequences without violating any operation constraints. Generally, the hot rolling scheduling contains three conflicting objectives: (1) product cost; (2) product quality; (3) roller use [4]. In order to ensure the surface quality and shape of the products, each roller needs replacement after rolled a certain number of slabs. Usually, the slab sequence between the two consecutive rollers is called a rolling round, and a rolling schedule contains a certain number of rolling rounds. Because of the high cost of roller replacement, the steel manufacturer should allocate as many slabs as possible to one rolling round within the capacity of rollers in order to reduce the product cost. At the same time, the slab sequence in a rolling round can affect the wearing of the roller.

Based on the practical experiences, the schedule with the width profile of “coffin shape” is the most favorable. Each rolling round contains two parts. The first part is called warm-up part, in which the slab width changes from narrow to wide in order to warm-up the roller. The warm-up part contains only a small amount of slabs. The second part is called the body part and most of slabs are processing in this part, in which the slab width changes from wide to narrow.

Although a rolling round is mainly scheduled according to the width profile, there are also several other constraints to be considered. It must follow the following constraints in the hot rolling scheduling in order to improve product quality, reduce roller wear and save production cost:

(1) Capacity constraints. Each rolling round has a length or weight limitation due to the roller capacity;
(2) The width sequence of warm-up part is from narrow to wide;
(3) The width sequence of body part is from wide to narrow;
(4) Jumps of rolling width, gauge and hardness must be smooth, and their maximum values must be less than a certain quantity;
(5) The amount of adjacent slabs with the same width has a limitation.

2.2. Objectives

There are several conflicting objectives in a HRSP. A good schedule should take into account the slab quality, roller replacement cost, smooth operation of the mill machine etc. The paper presents the objective function that consists of:

(1) Sequence-dependent transition penalty of width, gauge and hardness jump of adjacent slabs.
(2) Non-execution penalties. Each slab is given a production priority according to the customer’s demand, if one slab is not rolled in current rolling plan, it would bring a non-execution penalties.
2.3. Mathematical Modeling

From the above description, the HRSP can be summarized as follows. Supposed there are $N$ candidate slabs whose weight, width, gauge, hardness are known. Each rolling round has a certain capacity, and the adjacent slabs with the same width have a certain quantity limitation. The HRSP is to allocate these slabs into one or more rolling round and then sequence the allocated slabs in order to minimizing the integrated penalty by satisfying the above constraints.

Here we define the parameters and decision variables as follows.

**Parameters:**
- $i, j$: index of slabs
- $k$: index of rolling rounds
- $N$: total number of candidate slabs
- $K$: maximum number of rolling rounds
- $W_c$: maximum allowable number of slabs with same width within a round
- $W_l$: lower limit of roller capacity
- $W_u$: upper limit of roller capacity
- $w_i$: width of slab $i$
- $g_i$: gauge of slab $i$
- $h_i$: hardness of slab $i$
- $i_i$: weight of slab $i$
- $Q_i$: penalty for slab $i$ is not included in the schedule
- $d_i$: slab-priority factor of delivery due-date of slab $i$
- $p_{ij}^w$: penalty for width jump from slab $i$ to slab $j$
- $p_{ij}^g$: penalty for gauge jump from slab $i$ to slab $j$
- $p_{ij}^h$: penalty for hardness jump from slab $i$ to slab $j$
- $C_{ij}$: total penalty cost for parameter jump from slab $i$ to slab $j$, $C_{ij} = p_{ij}^w + p_{ij}^g + p_{ij}^h$

**Decision variable:**
- $X_{ijk}$: $= 1$ if slab $j$ is preceded by slab $i$ in round $k$
- $X_{ijk} = 0$ otherwise
- $Y_{ik}$: $= 1$ if slab $i$ is included in round $k$
- $Y_{ik} = 0$ otherwise
- $Z_{ijk}$: $= 1$ if slab $i$ and $j$ are all belong to round $k$ and rolling continuous with same width
- $Z_{ijk} = 0$ otherwise

In addition, a dummy slab $0$ which has no sequence-dependent transition penalty with other slabs is added to the set of manufacturing slabs as the starting node, and adjunctive constraints force the dummy to be included in scheduling sequence. The mathematical model of the proposed scheduling problem can be formulated as follows:

$$
\min \left\{ \alpha \cdot \sum_{k=1}^{K} \sum_{i=0}^{N} \sum_{j=0}^{N} C_{ij} \cdot X_{ijk} + \beta \cdot \sum_{i=1}^{N} Q_i \cdot (1 - \sum_{k=1}^{K} Y_{ik}) \right\} \tag{1}
$$

Subject to:

$$
\sum_{k=1}^{K} \sum_{j=0}^{N} X_{ijk} \leq 1, \quad i=1, \cdots, N, \quad k=1, \cdots, K \tag{2}
$$
\[
\sum_{k=1}^{K} \sum_{i=0}^{N} X_{ijk} \leq 1, \quad j = 1, \ldots, N, \quad k = 1, \ldots, K
\]  
(3)

\[
\sum_{i=1}^{N} X_{0jk} = 1, \quad k = 1, \ldots, K
\]  
(4)

\[
\sum_{j=1}^{N} X_{ijk} = 1, \quad k = 1, \ldots, K
\]  
(5)

\[
\sum_{j=1}^{N} Z_{ijk} Y_{jk} W_{f} \leq W_{c}, \quad i = 1, 2, \ldots, N, k = 1, 2, \ldots, K
\]  
(6)

\[
W_{f} \leq \sum_{i=1}^{N} W_{f} Y_{ik} \leq W_{c}, \quad k = 1, \ldots, K
\]  
(7)

\[
Z_{ijk} \leq Y_{ik}, \quad Z_{ijk} \leq Y_{jk}, \quad i, j = 1, 2, \ldots, N
\]  
(8)

\[
Y_{ik} \leq \sum_{j=0}^{N} X_{ijk}, \quad i, j = 1, 2, \ldots, N
\]  
(9)

\[
X_{ijk} \in [0,1], \quad Y_{ik} \in [0,1], \quad Z_{ijk} \in [0,1] \quad i, j = 0, \ldots, N, \quad k = 1, \ldots, K
\]  
(10)

Equation (1) is the objective function of the problem, and \( \alpha \) and \( \beta \) are the weights parameters for the relative importance of parts: total penalty cost for parameter jump and non-execution penalties of slabs. Constraints (2) and (3) denote that each slab is scheduled only once or not included in scheduling sequence respectively; Constraints (4) and (5) specify the dummy slab 0 as the starting node of scheduling sequence; Constraint (6) is the weight limitation of adjacent slabs with the same width; Constraint (7) is the upper bound and lower bound of roller’s rolling capacity; Constraint (8) is the groove constraint; Constraint (9) gives the relation between decision variables \( X_{ijk} \) and \( Y_{ik} \); Constraint(10) specifies that decision variables take only 0 or 1.

3. Population-based Extremal Optimization for HRSP

3.1. Extremal Optimization

Extremal optimization is a new stochastic method which origins from the model of self-organized criticality (SOC) in ecosystems [9]. Evolution in this model is driven by a process where the weakest species in the population, together with its nearest neighbors is always forced to mutate. Boettcher and Percus [11] have adapted the evolutionary model of [9] to tackle hard problems in combinatorial optimization, calling their algorithm extremal optimization. To improve performance and avoid a search based only on the forced mutation of the weakest species, they modified the basic EO algorithm introducing an adjustable parameter so that the search could escape local optima [12]. This variation of the EO algorithm is called \( \tau \)-EO and showed superior performance over the basic EO. EO and its derivatives have been extensively applied to solve numerous NP-hard combinatorial optimization problems. The simulation performance has been proved that EO outperforms other state-of-the-art algorithms in many applications, such as TSP [14], protein folding [15], and some industrial applications [13] [16].

3.2. Population-based Extremal Optimization for HRSP

EO exploits a single solution, with improvements achieved by repeatedly eliminating those components producing the worst fitness. EO algorithm has very strong local search ability but uses only one solution that can easily miss some promising area in the searching space. Chen et al. [17] presented a population-based extremal optimization algorithm, but it only was applied to solve continuous constraints
optimization problem, besides the solutions of population don’t share information with each other during the evolution process.

By introducing the population search strategies which is popularly used in evolutionary algorithms, such as GA and PSO to EO, we develop a new algorithm, called population-based extremal optimization algorithm (PEOA), for solving the HRSP. The computation flow of PEOA can be shown as follows:

**Step 1.** The optimization process starts from generating initial population with $m$ solutions with heuristics method. Let $S_{best}$ represent the best solution found so far, and then choose one solution with the best performance from the initial population as $S_{best}$.

**Step 2.** For each solution $S$ in the current population:
(a) Evaluate localized fitness $\lambda_i$ for each variable $x_i$
(b) The components (variables) of solution $S$ are ranked according to their localized fitness, using a rank $r = 1$ for the worst variable and $r = l$ for the best, where $l$ is the variable quantity of solution $S$.
(c) A candidate variable $x_i$ is chosen for mutation and confirmed with a probability: $P(i) = r^{-\tau}$, where $r$ is the localized fitness rank of variable $x_i$, and the power-law exponent $\tau$ is an adjustable parameter.
(d) For updating the state of the selected variable $x_i$, we construct the neighborhood $N(S)$ of solution $S$, and then choose the best solution $S' \in N(S)$ such that the variable $x_i$ must change its state.
(e) If $C(S') < C(S_{best})$, then set $S_{best} = S'$, where $C(S)$ is the fitness of solution $S$.
(f) Accept $S \leftarrow S'$ unconditionally.

**Step 3.** Population evolution strategy: construct a number of new solutions by combination method and replace those worst ones at specified generation intervals.

**Step 4.** Loop to Step 2 until termination criterion is met, which is either a certain number of generations or a predefined amount of CPU time.

**Step 5.** Output the results.

In order to apply the PEOA to solve the HRSP, we have given several new definitions on the species and localized fitness of the problem. The detailed discussion of applying EO algorithm is shown as follows.

### 3.2.1. Representation of Solutions and Fitness Function

The solution represents the production sequence, and the sequenced slabs marked with slab ID respectively. Assuming there is a HRSP with $N$ candidate slabs and $K$ rolling rounds, we use a permutation of integers to encode the solution. The integers from 1 to $N$ represent the candidate slabs, and the integers $N+1, N+2, \ldots, N+K$ represent $K$ dummy slabs that separate different rolling rounds in the whole rolling schedule. With this representation scheme, the information on both the slabs and the rolling rounds are embedded in a single solution structure.

Fitness function is used to evaluate the performance of solutions. In the paper, the objective function of solution $S$ is defined as:

$$\text{Fit}(S) = \alpha \sum_{k=1}^{K} \sum_{i=0}^{N} \sum_{j=0}^{N} C_{ij} \cdot X_{kj} + \beta \sum_{i=1}^{N} \left( \sum_{k=1}^{K} Y_{ik} \right) + W \times \text{feas}_S \quad (11)$$

where:

$$\text{feas}_S = \begin{cases} 0 & \text{if } S \text{ is feasible} \\ 1 & \text{otherwise} \end{cases} \quad (12)$$

### 3.2.2. Population Initialization

Generally, there are two issues to be considered for population initialization in population-based evolutionary algorithms: the initial population size and the procedure to initialize the population [4]. A large population size is faced with excessive time complexity and high computational cost; on the contrary, a
small quantity of individuals cannot efficiently locate the optimal solution. So, determining an appropriate population size is crucial to find the optimal scheduling solution. The population size is set as a default number $pop = K \times 50$ in this application, where $K$ is the amount of rounds in the schedule.

Furthermore, heuristic initialization and random initialization are among the most popular ways to generate the initial population. Heuristic methods can create sub-optimal solutions with high mean fitness. We take the initialization procedure proposed in literature [4] as population initialization method.

3.2.3. Species and Localized Fitness

Each solution is composed of a certain number of variables in the EO method. Each variable is considered as one species and to each of them a localized fitness is assigned that represents the level of adaptability of that species. To solve the HRSP by the EO algorithm, each slab or dummy slab of a solution can be seen as a species, and then a localized fitness is assigned to each slab. After taking into account the width, thickness and hardness jump penalty between adjacent slabs and the penalty when the slab is not included into the scheduling sequence, the localized fitness of each slab can be defined as:

$$l_i = \alpha (C_{pr(i)} + C_{po(i)}) - \beta \cdot Q_i$$

(13)

where $pr(i)$ and $po(i)$ respectively, represent the predecessor and successor of slab $i$ in scheduling sequence, and $Q_i$ represent the penalty when slab $i$ is not included into the scheduling sequence.

3.2.4. Neighborhood Function

The neighborhood function of Step 2. d) is to updating the state of the selected species, and we can design different neighborhood functions according to actual scheduling problems. For solving the HRSP, we construct the neighborhood $N(S)$ of solution $S$ by the route-improvement method [5].

3.2.5. Population Evolution Strategy

The PEOA also has information exchange between different solutions during searching process, just like other population-based searching techniques such as GA [4], PSO [18], etc. The population evolution strategy in Step 3 can be described as:

1. At specified generation intervals during the searching process, select $m$ best solutions from the current population according to their fitness function.
2. Separate these selected solutions into different sets that will be used for creating new solutions with the solution combined method. In order to increase variety of new solutions, the separate method is organized to generate two different types of solution set which are shown as $R_1$ and $R_2$:
   - $R_1$: Generated by pairs of solutions from the $m$ best solutions.
   - $R_2$: Generated by three solutions from the $m$ best solutions.

Use solution combination method to create new solutions and replace those worst solutions in the population. Solution combination method uses each generated solution set and combines the solutions in it, returning one or more solutions. In the paper, we take the solution combination method in literature [19] to generate new solutions.

4. Computational Results and Comparison

The algorithm is programmed in Visual C++ 6.0 and run on the Intel Pentium 2.8G with 512M RAM. The maximum of iterative generations is set to $K \times 100$ in this application and population size is $pop = K \times 50$, where $K$ is the amount of rounds in the schedule. The parameter $\tau$ of EO is set to 1.2.

To evaluate the performance of the PEOA, extensive experiment tests have been performed with production scale data. The scheduling objective is to minimize the total penalty cost. A typical structure for width, hardness and gauge changeover cost which is proposed by Kosiba et al. [9].
4.1. A Case Study

Firstly, for a set of slabs (30 warm-up slabs and 350 body part candidate slabs) processed in one rolling round, the PEOA give the rolling schedule satisfied the constraints, which is shown in Fig 1. It can be seen that the distribution pattern of slabs’ width has typical “coffin shape”, and the gauge and hardness changes smoothly and then the schedule is a reasonable rolling sequence.

4.2. Comparison between PEOA and Other Algorithm

In order to illustrate the performance of PEOA, we compare the results of PEOA from other algorithms, such as manual scheduling, EO algorithm, MGA [4], etc. The termination conditions of these algorithms are set equal to the PEOA’s CPU time. The statistical performance of 20 independent runs of these algorithms are listed in table 1, where $M$ is the quantity of warm-up candidate slabs, $N$ is the quantity of body part candidate slabs, $K$ is the quantity of rolling round, Best is the best objective value after 20 runs, $\eta$ is the average improvement percentage over manual scheduling after 20 runs, $t$ is the average CPU time of PEOA over 20 runs.

It can be seen that the PEOA developed in this paper can further reduce the objective function value and provide a more favourable scheduling solution. The superiority of the best optimization quality demonstrates the effectiveness and the global search property of the PEOA, and the superiority of the average performance over 20 random runs shows that the PEOA is more robust than EO and MGA. For more large scale problem, we can take a trade-off between computation time and solution quality by adjusting the quantity of generations or the population scale of the PEOA.

Table 1 Comparison on Scheduling Results of Different Algorithms

<table>
<thead>
<tr>
<th>Instance</th>
<th>$M$</th>
<th>$N$</th>
<th>$K$</th>
<th>Manual scheduling</th>
<th>EO Best</th>
<th>EO $\eta$ (%)</th>
<th>EO Best</th>
<th>EO $\eta$ (%)</th>
<th>EO Best</th>
<th>EO $\eta$ (%)</th>
<th>EO $t$</th>
<th>MGA Best</th>
<th>MGA $\eta$ (%)</th>
<th>MGA Best</th>
<th>MGA $\eta$ (%)</th>
<th>PEOA Best</th>
<th>PEOA $\eta$ (%)</th>
<th>PEOA $t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
<td>350</td>
<td>1</td>
<td>486</td>
<td>471</td>
<td>1.28</td>
<td>471</td>
<td>1.57</td>
<td>471</td>
<td>1.76</td>
<td>15.93</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>70</td>
<td>580</td>
<td>2</td>
<td>978</td>
<td>941</td>
<td>2.21</td>
<td>936</td>
<td>2.52</td>
<td>932</td>
<td>2.64</td>
<td>45.83</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>750</td>
<td>3</td>
<td>1436</td>
<td>1378</td>
<td>2.72</td>
<td>1362</td>
<td>3.41</td>
<td>1345</td>
<td>3.98</td>
<td>159.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>175</td>
<td>1050</td>
<td>4</td>
<td>2048</td>
<td>1957</td>
<td>3.85</td>
<td>1931</td>
<td>4.25</td>
<td>1920</td>
<td>5.04</td>
<td>398.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Conclusion

In the paper, we present a general description for the HRSP and then a mathematical model is proposed to formulate it. Since this problem is hardly possible to be solved with exact optimization methods, an effective evolutionary algorithm which is combined EO with population evolution techniques, called PEOA, is developed to solve the problem. The experimental results with real production data show that the solutions provided by the proposed PEOA have out-performed other algorithms such as EO, MGA, etc. Moreover, the idea presented here can be easily adapted to solve other industrial problems.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No. 60574063

References