Computers & Operations Research 39 (2012) 339–349

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Development of hybrid evolutionary algorithms for production scheduling of hot strip mill

Yu-Wang Chen [a,c,*], Yong-Zai Lu [a], Ming Ge [b], Gen-Ke Yang [a], Chang-Chun Pan [a]

[a] Department of Automation, Shanghai Jiaotong University, Shanghai 200240, PR China
[b] Honeywell Technology & Solutions Lab, Shanghai 201203, PR China
[c] Decision and Cognitive Sciences Research Centre, MBS, The University of Manchester, Manchester M15 6PB, UK

ARTICLE INFO

Available online 21 April 2011

Keywords:
Hot strip mill
Production scheduling
Hybrid evolutionary algorithm
Genetic algorithm
Extremal optimization

ABSTRACT

A hot strip mill (HSM) produces hot rolled products from steel slabs, and is one of the most important production lines in a steel plant. The aim of HSM scheduling is to construct a rolling sequence that optimizes a set of given criteria under constraints. Due to the complexity in modeling the production process and optimizing the rolling sequence, the HSM scheduling is a challenging task for hot rolling production schedulers. This paper first introduces the HSM production process and requirements, and then reviews previous research on the modeling and optimization of the HSM scheduling problem. According to the practical requirements of hot rolling production, a mathematical model is formulated to describe two important scheduling sub-tasks: (1) selecting a subset of manufacturing orders and (2) generating an optimal rolling sequence from the selected manufacturing orders. Further, hybrid evolutionary algorithms with integration of genetic algorithm (GA) and extremal optimization (EO) are proposed to solve the HSM scheduling problem. Computational results on industrial data show that the proposed HSM scheduling solution can be applied in practice to provide satisfactory performance.

1. Introduction

A hot strip mill (HSM) produces hot rolled products from steel slabs, and is one of the most important production lines in an integrated mill or mini-mill [1,2]. Besides the process control strategies such as automatic gauge control and rolling force prediction, production planning and scheduling also significantly affects the performance of a hot strip mill in terms of product quality, throughput and on-time delivery. As a consequence, HSM scheduling has become a critical task in a steel plant. Generally speaking, the primary aim of a HSM scheduling solution is to generate an optimal production sequence consisting of a single rolling round or multiple consecutive rounds (or so-called campaign). A rolling round must optimize a set of given criteria and satisfy a series of constraints, such as the "coffin shape" of width profile, the smooth jumps in dimensions and hardness between adjacent coils, the minimal and maximal number of coils or footages, the maximal number of coils with the same width, the maximal number of short slabs, work-in-process (WIP) inventory and on-time delivery.

Due to the large number of manufacturing orders, multiple conflicting objectives and various production constraints, the HSM scheduling problem has been proven to be a typical non-deterministic polynomial (NP)-hard combinatorial optimization problem [3–5]. It is almost impossible to generate an optimal scheduling solution by human schedulers or traditional mathematical programming methods. Balas [3] formulated the round scheduling problem as a generalized traveling salesman problem (TSP) with multiple conflicting objectives and constraints. Kosiba et al. [6] stated that the minimization of roller wears is equivalent to the composite objective of improving product quality, production rate and profits, and the roller wears can be measured by a penalty function which is determined by the jump values in width, gauge and hardness between adjacent coils. Assaf et al. [7] incorporated four key sub-models (i.e., rolling mill, reheat furnace, heat loss and cost calculation), and developed an enumeration based branching and pruning algorithm to generate scheduling solutions for the steel production sequence problem. Lopez et al. [4] presented an aggressive heuristic to solve the HSM scheduling problem. The heuristic repeatedly applies the tabu search method and the CROSS exchange operator in the cannibalization stage. Tang and Wang [8] modeled the hot rolling production scheduling problem as a prize collecting vehicle routing problem (PCVRP), for which an iterated local search algorithm (ILS) was proposed on the basis of very large-scale neighborhood (VLSN) using cyclic transfer. Due to the complexity

* Corresponding author at: Department of Automation, Shanghai Jiaotong University, Shanghai 200240, PR China. Tel.: +86 21 3420 5284.
E-mail addresses: cywpeak@gmail.com, Yu-wang.chen@mbs.ac.uk (Y.-W. Chen), y.lu@ieee.org (Y.-Z. Lu).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.04.009

of the scheduling models and the inefficiency of mathematical programming methods, various intelligent methods, including local search and greedy algorithm [9–11], genetic algorithm [12–14], tabu search [15] and particle swarm optimization [29] have been widely applied in past decades to solve the HSM scheduling problem. Furthermore, scheduling systems with different features have also been developed. Cowling [16] introduced a semi-automatic decision support system featured with a multi-objective scheduling model, and the model was solved by a variety of bespoke local search and tabu search methods. Knoop and Van Nerom [17] proposed a scheduling architecture for an integrated steel plant. In the IBM research reports, the overall scheduling in steel manufacturing process was decomposed as primary production scheduling [1] and finishing line scheduling [18].

By referring to the existing scheduling models, algorithms and systems in steel industry, this study explores the development of a practical HSM scheduling solution. A mathematical model is first formulated to describe two important scheduling sub-tasks: (1) selecting a subset of manufacturing orders and (2) generating an optimal rolling sequence from the selected manufacturing orders. Hybrid evolutionary algorithms with integration of genetic algorithm (GA) and extremal optimization (EO) are further proposed to solve the HSM scheduling problem in a practical way.

The rest of this paper is organized as follows: Section 2 introduces the HSM production process and its scheduling objectives and constraints; Section 3 presents a mathematical scheduling model; Section 4 focuses on solving the HSM scheduling problem by proposing hybrid evolutionary algorithms and developing a practical scheduling system. Computational results on industrial data are reported in Section 5. Finally, Section 6 concludes this paper.

2. Problem statement

2.1. Production process

Usually, a hot rolling production line consists of reheating furnaces, a set of roughing mills and finishing mills, a water-cooler and a coiler as schematically shown in Fig. 1.

In a hot rolling mill, the steel slabs from continuous casters or slab yards are first charged to a working beam or push type reheating furnace, and then the heated slabs discharged from the reheating furnace are processed through a set of roughing stands which make use of horizontal rollers to reduce the slab thickness (initially 20–30 cm) and vertical rollers to regulate the slab width. Subsequently, the intermediate slabs are loaded into a finishing mill, in which 6–7 stands can further reduce the gauge and width of slabs to desired values. After passing through the water-cooler and the coiler, the raw slabs are finally converted to the finished hot-rolled products with desired gauge (1.5–12 mm), width, mechanical and thermal properties, and surface quality.

2.2. Scheduling objectives and constraints

Production scheduling, in general terms, is the process of resource distribution and order arrangement for a period of time under a set of given criteria and constraints. In hot rolling production, as discussed above, the primary aim of a scheduling solution is to construct an optimal production sequence which can simultaneously finish two sub-tasks: (1) selecting a subset of manufacturing orders with the "make-to-inventory" or "make-to-order" principle and (2) generating an optimal rolling sequence. Effective HSM scheduling is crucial to a steel plant's competitiveness, since the hot rolling mill is one of the most important production lines in steel industry [4]. In a hot rolling mill, the finished products are produced by subjecting steel slabs to high pressures through a series of rollers. As a consequence, the working and backup rollers need to be replaced periodically due to abrasions. Usually, the set of slabs rolled between two consecutive replacements of working rollers is called a *round*, and the set of slabs rolled between two consecutive replacements of backup rollers is called a *campaign*. Fig. 2 illustrates the empirical "coffin-shape" width profile of a rolling round and the multiple rolling rounds of a campaign. As shown in the figure, a coffin-shape rolling round mainly consists of two parts: the *warm-up* section, in which several slabs (5–15) are rolled from narrow to wide for warming up the rollers, and the *body* section, in which a number of slabs are rolled from wide to narrow for avoiding marks or grooves at the edge of rolled coilers.

In terms of their relative importance, the HSM scheduling constraints can be classified into hard constraints and soft constraints. Hard constraints refer to those that cannot be violated in the final scheduling solution. In the HSM scheduling problem, the main hard constraints are as follows:

(1) A rolling round has a "coffin shape" width profile, which starts with a warm-up section having the width pattern of narrow to wide, and follows a body section having the width pattern of wide to narrow.
(2) The number of coils in a rolling round has lower and upper bounds due to the capacity of the rollers.
(3) The changes in dimensions and hardness between adjacent coils should be smooth in both warm-up and body sections. This is because the rollers need to be adjusted for controlling the dimension and hardness jumps on the entry and outlet of mills.
(4) The number of coils with the same width to be processed consecutively has an upper bound in a rolling round. Otherwise the edges of slabs mark the rollers easily. In practice, this is also called groove constraint.

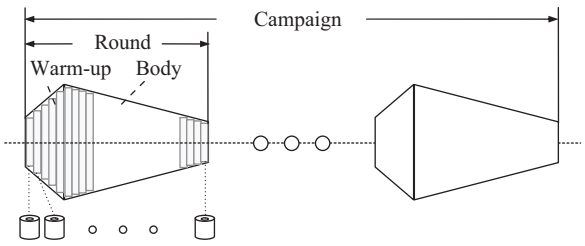

Fig. 2. "Coffin-shape" width profile of rolling round and the composition of campaign.

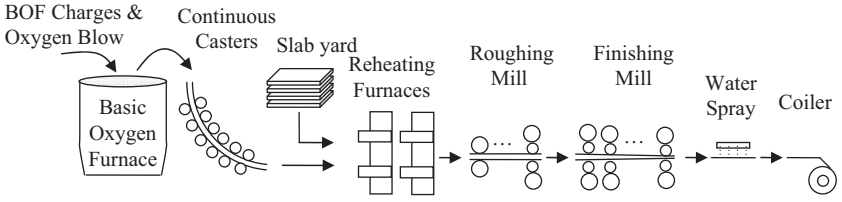

Fig. 1. Hot rolling production line.

(5) The number of short slabs in a rolling round has an upper bound, because reheating furnaces are generally designed for normal slabs and short ones are only used to fill in gaps.

Soft constraints, on the contrary, can be compromised and violated according to their priorities. In this paper the following soft constraints are considered for generating rolling rounds:

(1) To provide satisfactory delivery service and maintain lower work-in-process (WIP) inventory, the hot-rolled products should be processed near its due date, and not too early or too late for decreasing earliness/tardiness (E/T) penalties.
(2) The hardness and desired surface quality of the coils processed in a warm-up section should not be too high.
(3) The coils with high quality requirements should not be processed in the rear part of a body section, because the performance of rollers degrades with the increase of rolling time.

In the HSM scheduling, there are a number of conflicting objectives to be considered. For example, product quality and production cost are two of the most common conflicting objectives. To improve the product quality, the rollers need to be replaced more frequently, which incurs extra production cost and decreases the rate of throughput. Similarly, on-time delivery is in conflict with low WIP inventory. Therefore, an effective scheduling system should be capable of making tradeoffs among multiple conflicting objectives. In the HSM scheduling, the optimization objective is defined as the weighted combinations of the penalties for width, gauge, hardness jumps between adjacent coils, the penalties for the violation of short slab and groove constraints and the due-date E/T penalties [4,14]. Generally, the HSM scheduling system is situated in a dynamic environment. The manufacturing orders are only known for a limited period, and new ones transformed from customer orders are downloaded from upper-level enterprise resource planning (ERP) and production planning system periodically. This leads to the HSM scheduling being a periodic process [15]. Therefore, we can schedule rolling rounds one by one, and the HSM scheduling can be simplified to generate an optimized rolling round by selecting and sequencing a subset of coils from existing manufacturing orders.

3. Problem formulation

This section presents a mathematical HSM scheduling model. Two scheduling sub-tasks as discussed above are considered simultaneously. Traditionally, the HSM scheduling problem is solved by using a sequential strategy, which first selects a subset of manufacturing orders and then searches the optimal rolling round within the selected orders. This strategy, although straightforward, can only find a local optimum for the round scheduling. In this paper, the proposed scheduling model formulates the two coupling sub-tasks in an integrated way, and hence is beneficial to find the globally optimal scheduling solution. First of all, the scheduling solution is formulated as a directed graph $G(V,A)$, where the node set $V=(1,2,\ldots,N)$ represents N manufacturing orders (coils) with desired attributes and due-date, and the arc set denotes the transition trajectories between coils i and j. For each arc (i,j), let c_{ij} denote the sequence-dependent penalty incurred by rolling coil j immediately after coil i. As shown in Table 1, the transition penalty can be measured according to the jump values in dimensions and hardness between adjacent coils.

To enhance the accuracy of the penalty structure, the jump values are scaled for different width or gauge rates. Undoubtedly, a gauge jump of 0.15 mm for 1.5 mm coils is more difficult to manipulate than the same jump for 7.5 mm coils.

To formulate a mathematical scheduling model, the parameters are defined as follows.

i,j	number of coils
N	total number of coils to be produced for manufacturing orders
w_i, g_i, l_i,h_i	width, gauge, length and hardness of coil i
u_i	finished temperature of coil i in the outlet of finishing mill
s_i	logic symbol of short slabs, i.e., if the raw slab of producing coil i is shorter than a predefined length, $s_i=1$, otherwise, $s_i=0$
V	width set of all coils to be produced, $V=\{\upsilon_1,\upsilon_2,\ldots,\upsilon_k\}$ $(k\le n)$
N_υ	maximal number of coils with the same width to be processed consecutively in a rolling round
N_s	maximal number of short slabs in a rolling round
L_l,L_u	minimal and maximal lengths of coils in a rolling round
c_{ij}	transition cost for rolling coil j immediately after coil i
c_{ii}	cost for not selecting coil i to the current rolling round
d_i	due-date of coil i
p_i	processing time of coil i
t_i	starting time of processing coil i
T_s	starting time of the current rolling round

We also define the following decision variables:

$$x_{ij}=\begin{cases}1 & \text{if coil } j \text{ immediately follows coil } i\\ 0 & \text{otherwise}\end{cases}$$

Since this paper aims to solve the two sub-tasks of order selection and sequencing simultaneously, another decision variable x_{ii} is defined below by analogy with the prize collecting TSP model [4].

$$x_{ii}=\begin{cases}1 & \text{if coil } i \text{ is not selected in the current rolling round}\\ 0 & \text{otherwise}\end{cases}$$

The sequence-dependent transition cost c_{ij} can be defined as $c_{ij}=p_{ij}^w+p_{ij}^g+p_{ij}^h+p_{ij}^t$, where p_{ij}^w, p_{ij}^g, p_{ij}^h and p_{ij}^t represent the width,

Table 1
Transition penalty for the changes in width, gauge and hardness.

Width range (cm)	Inward jump (cm)	Penalty	Gauge range (mm)	Jump value (mm)	Penalty	Hardness jump	Penalty
85–100	0–3	1	1.2–3.0	0–0.03	3	1	5
85–100	3–6	3	1.2–3.0	0.03–0.06	7	2	15
85–100	6–9	10	1.2–3.0	0.06–0.09	15	3	35
…	…	…	…	…	…	4	50
85–100	50–	500	1.2–3.0	0.45–	200	5	75
…	…	…	…	…	…	…	…
150–	50–	1000	12.0–	0.45–	1000	9	1000

gauge, hardness, and finished temperature transition cost respectively. As discussed above, the transition cost can be measured by the jump values of the parameters of every two adjacent coils. For example, the finished temperature transition cost can be measured by a function $p_{ij}^t = p^t(u_j - u_i)$. In practice, p_{ij}^w, p_{ij}^g and p_{ij}^h are usually defined by the penalty structure as shown in Table 1.

In the HSM scheduling, the optimization objective can be decomposed into two parts according to the information requirements of evaluating fitness [19]: (1) local evaluation function (LEF), which can be calculated by using only local information. The LEF is defined as $LEF(S) = \sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ij}$, which consists of the sequence-dependent transition costs and the non-execution penalties [20] and (2) global evaluation function (GEF), which needs to consider the overall configuration information of a solution. The E/T penalties can be included in this part, and therefore the GEF is defined as $GEF(S) = \sum_{i=1}^{n}(e_i + r_i)$, where $e_i = \max\{0, d_i - t_i - p_i\}$ and $r_i = \max\{0, t_i + p_i - d_i\}$. To generate a rolling sequence, a virtual coil is added to the set of manufacturing orders, and it has no processing time and any sequence-dependent transition cost. A constraint is added to select the virtual coil as the starting node of a rolling round. As a result, a mathematical HSM scheduling model can be formulated as follows:

Minimize

$$\lambda \sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ij} + (1-\lambda)\sum_{i=1}^{n}(e_i + r_i) \tag{1}$$

Subject to

$$\sum_{i=0}^{n} x_{ij} = 1, \quad j = 0,\ldots,n \tag{2}$$

$$\sum_{j=0}^{n} x_{ij} = 1, \quad i = 0,\ldots,n \tag{3}$$

$$L_l \le \sum_{i=0}^{n} l_i(1-x_{ii}) \le L_u \tag{4}$$

$$\sum_{i=0}^{n}(1-x_{ii}) * \mathrm{sgn}(w_i - \upsilon_k) \le N_\upsilon, \quad \forall \upsilon_k \in V \tag{5}$$

$$\sum_{i=0}^{n} s_i(1-x_{ii}) \le N_s \tag{6}$$

$$x_{00} = 0 \tag{7}$$

$$t_0 = T_s \tag{8}$$

$$t_j = \sum_{i=0}^{n} x_{ij}(t_i + p_i) + x_{jj}(t_0 + T), \quad j = 1,\ldots,n \tag{9}$$

$$x_{ij} \in \{0,1\}, \quad i,j = 0,\ldots,n \tag{10}$$

$$\mathrm{sgn}(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Formula (1) is used to calculate the optimization objective $F(S)$ of any scheduling solution S, and the parameter $\lambda (0 \le \lambda \le 1)$ is the weight of measuring the relative importance of the two optimization parts. Constraints (2) and (3) indicate that each coil can be processed only once or it is not selected in the current rolling round. Constraints (4) specifies the minimal and maximal capacities of a rolling round. Constraints (5) and (6) represent the maximal number of same-width coils and that of short slabs, respectively. Equalities (7) and (8) select the virtual coil as the starting node of a rolling round. Eq. (9) establishes the relationship between variables t_j and x_{ij}, where constant T is greater than the total processing time of a rolling round. It means that if coil j is not selected in the current rolling round, its processing time is not earlier than the starting time of the next rolling round.

4. Hybrid optimization method for HSM scheduling

Considering the characteristics of the HSM scheduling problem, which is a typical NP-hard problem with a composite optimization objective, it is unrealistic to search for the optimal scheduling solution by using traditional mathematical programming methods. Therefore, this paper develops a hybrid optimization method as schematically illustrated in Fig. 3. A heuristic-based approach is designed to generate the warm-up section, and hybrid evolutionary algorithms combining GA and EO are developed to optimize the body section.

4.1. A heuristic-based approach for scheduling the warm-up section

A warm-up section can be scheduled easily, since it only consists of a few number of slabs (5–15) for warming up the rollers. The main requirement of this section is that coils must be rolled from narrow to wide. In this section, a heuristic-based approach, which selects rolling coils from manufacturing orders uniformly, is proposed as follows.

Algorithm 1 (*Pseudo-code for scheduling the warm-up section*)

Step 1: Identify available slabs which can be processed in a warm-up section within the current scheduling time horizon.

Step 2: If the number of available slabs is less than the required number $N_{warm\text{-}up}$ for a warm-up section, the scheduling system suggests downloading more manufacturing orders from upper-level planning systems, then go to *Step* 6.

Step 3: Sequence all corresponding coils in ascending width and due-date.

Step 4: Group the coils with the same width, and calculate the number of groups N_w.

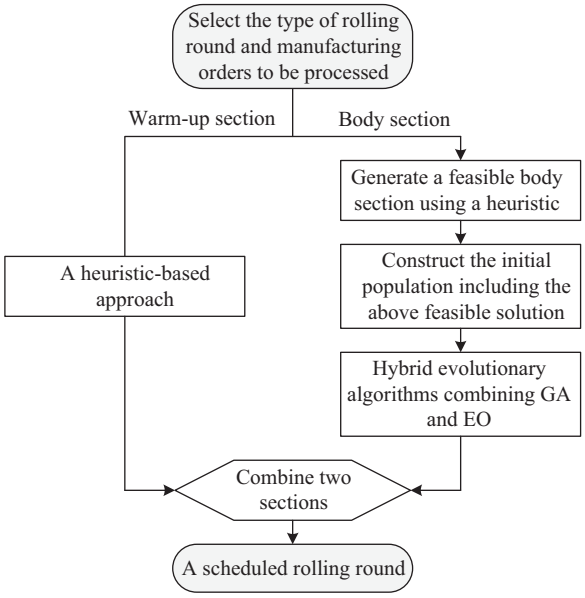

Fig. 3. Scheme of the hybrid optimization method for HSM scheduling.

Step 5: Select the first coil of each width group into a warm-up set successively, and delete the selected coils from the corresponding width group.
If $N_{warm\text{-}up} > N_w$, select all coils in the warm-up set into a provisional warm-up section;
Select one coil from the width group with the maximal residual coils, and insert it into the warm-up set according to the width profile from narrow to wide. Repeat this procedure until $N_{warm\text{-}up}=N_{temp}$, then go to *Step* 6;
Else, generate a warm-up section by sequentially selecting the coils in the warm-up set with the step size $\lfloor N_w/N_{warm\text{-}up} \rfloor$.
Step 6: Stop the selection for the warm-up section, and return the scheduled output.

The warm-up section generated by the above heuristic-based approach has a gradually increasing width pattern as described in Section 2.2.

4.2. Hybrid evolutionary algorithms for scheduling the body section

The body section is the kernel part of a rolling round. The performance of a HSM scheduling solution mainly depends on the optimization quality of the rolling sequence of the body section. Without specific explanation, the scheduling of the body section is usually regarded as the equivalent of the HSM scheduling as stated in many papers [4,14]. The mathematical model formulated in Section 3 describes the scheduling of the body section, and it is a NP-hard combinatorial optimization problem. Therefore, complete enumeration of all possible solutions is computationally prohibitive, i.e., no deterministic algorithm is capable of solving the HSM scheduling problem within reasonable computation time. As one class of effective optimization techniques, intelligent algorithms have been widely employed to solve typical scheduling problems for effectively finding a desirable solution, although it may not necessarily be the optimal solution [2]. In this paper, an effective hybrid evolutionary algorithm is developed to solve the HSM scheduling problem by combining the population-based search capacity of GA and the fine-grained local search efficacy of EO.

4.2.1. Global search algorithm: modified genetic algorithm for order selection and sequencing

Genetic algorithm is inspired by the Darwinian evolution theory [21]. It is a class of population-based search methods, and the iterative improvement is realized by a generate-test procedure [19]. In past decades, the genetic algorithm has been widely applied to the field of combinatorial optimization. In this section, a modified GA (MGA) is proposed to solve the HSM scheduling problem.

4.2.1.1. Representation of solutions. Generally, standard GA applications use binary strings or ordinal integers to represent a chromosome of a solution. In the HSM scheduling problem, we define the chromosome with a chain of genes as a rolling round. Each gene represents a coil marked with Coil ID. Suppose that the number of coils in a chromosome is equal to m and the total number of coils in the order book is n, the scheduling objective is to select m coils from the n coils and generate an optimized rolling sequence. An example of a chromosome is shown in Fig. 4.

The vector [5, 8, …, 3] denotes a possible rolling round, and each figure in the vector represents a particular coil ID. This chromosome can represent a possible scheduling solution in which the rolling round consists of 9 coils, but the total number of coils can be more than 12.

Slot#:	1	2	3	4	5	6	7	8	9
Chromosome:	5	8	9	12	7	11	2	10	3

Fig. 4. Illustration of chromosome representation.

4.2.1.2. Population initialization. Two issues are usually discussed for the population initialization: the population size and the method of initializing the population [5,22]. To maintain the searching capability, the population size needs to increase exponentially with the dimension of the optimization problem, i.e., the length of the chromosome. However, although a large population size can search the solution space effectively, it also incurs excessive computing time. As a result, an appropriate population size is crucial for finding the optimal solution efficiently and effectively.

In a large number of GA applications, heuristic and random methods are among the most popular approaches to generate the initial population. Well-designed heuristic methods can be used to produce some good individuals, which is beneficial to improve the convergence speed. However, if all initial individuals have good fitness values, the algorithm converges to locally optimal solutions easily and never explores the whole solution space due to lack of genetic diversity. On the contrary, if all initial individuals are generated randomly, it may take a large number of generations to improve the inferior individuals. A random initial population makes it difficult to obtain a good solution, particularly for a practical application with constraints. Therefore, in the proposed MGA, three different methods are employed simultaneously to generate the initial population.

First of all, a well-designed heuristic is proposed to generate a feasible solution. The constraints discussed in Section 2.2 are considered in this heuristic. The initial feasible individual can be used to improve the feasibility of all iterative-generated individuals and accelerate the convergence speed.

Algorithm 2 (*Pseudo-code for the heuristic to generate a feasible body section*)
Step 1: Identify available slabs for the body section within this current scheduling time horizon.
Step 2: Sequence the corresponding coils in descending width and ascending due-date (i.e., the coils are first sequenced in descending width, and those with the same width are further sequenced in ascending due-date).
Step 3: Group the coils with the same width, and calculate the number of groups N_w.
Step 4: Select N_v coils into a body set from each group sequentially, and calculate the number of selected coils N_{temp} in the body set.
Step 5: If N_{temp} is less than the required number N_{body} for a body section, it means no feasible body section can be generated with existing manufacturing orders, then go to *Step* 6;
Else, generate a body section by sequentially selecting the coils in the body set with the step size $\lfloor N_{temp}/N_{body} \rfloor$.
Step 6: Stop the heuristic, and return the scheduling output.

Secondly, the nearest neighbor search method, which first chooses a starting coil and then selects the next least-cost coil to the provisional sequence iteratively, is used to generate a proportion of individuals in the initial population.

Finally, the random insertion method is used to generate all other individuals. Starting from a randomly selected coil, a body section is generated by selecting the next unscheduled coil randomly and then inserting it the least-cost position of the provisional sequence.

4.2.1.3. Fitness function. The fitness function calculates how fit an individual is, and the "fittest" ones have more chances to be inherited into the next generation. In the HSM scheduling problem, the fitness is defined as a composite optimization objective as discussed in Section 3.

4.2.1.4. Genetic operators

(A) *Selection*: The selection operator is used to improve the mean fitness values of the population by giving the better chromosomes higher probabilities to pass their genotypic information to the next generation. The selection schemes are usually characterized by a selection pressure, which is defined as the ratio of the selection probability of the best chromosome to that of an average chromosome. The rank-based selection scheme selects individuals based on the rank of their fitness values, and the selection probability is defined as $p_i = c(1-c)^{i-1}$, where c denotes the selection pressure, and i is the rank number of a chromosome in the whole population [21].
To generate new individuals, one parent is chosen from the feasible solution pool by the roulette-wheel method, and another parent is selected from the current population using a niche technique. The niche technique ensures a minimum difference between every two parent chromosomes and further maintains the genetic diversity of the new population. Thus, a similarity coefficient is defined as $c_{ij} = s_{ij}/n$, where n is the chromosome size, s_{ij} is the number of identical genes between chromosomes i and j. When a parent chromosome i is selected, only the chromosome j, whose similarity coefficient c_{ij} with the chromosome i is not higher than a predetermined value c_0, has the possibility to be selected as the second parent chromosome.

(B) *Crossover*: The crossover operator transforms parent chromosomes for finding better child chromosomes. Partially Matched Crossover (PMX) and Ordered Crossover (OX) have been proved as effective crossover schemes for integer chromosomes [23]. In the HSM scheduling problem, the integral range is equal to, or greater than the chromosome length. As a result, the genes in a child chromosome are not completely homologous to that of its parents, and the crossover operator in the MGA is also not completely identical to that of standard PMX and OX. Here, we present a simple example to illustrate the PMX operator in the MGA. Given two parent chromosomes S_1 and S_2:

$S_1 : 5-8-9-|12-7-11|-2-10-3,$
$S_2 : 7-6-11-1-9-10-5-4-8$

First, two cut points are chosen at random, and the genes bounded by the cut points are exchanged. As a result, one chromosome has some new partial genetic information from the other. The intermediate structures of the new solutions are

$S_1' : 5-8-9-|1-9-10|-2-10-3,$
$S_2' : 7-6-11-|12-7-11|-5-4-8$

However, these two intermediate solutions are not necessarily feasible because some genes are repeated. The repeated genes can be replaced by mapping $|12-7-11|$ to $|1-9-10|$. And then two new solutions are generated as follows:

$S_1' : 5-8-7-|1-9-10|-2-11-3,$
$S_2' : 9-6-10-|12-7-11|-5-4-8$

One can see that the gene 1 is not included in the parent chromosome S_1, but it appears in its child chromosome S_1' after the PMX operation.

(C) *Local search as mutation*: The mutation operator generates a new chromosome from a selected one. In the MGA, the *Or-opt* exchange is employed as the mutation operator. It is one of the chain exchanging methods and attempts to improve the current solution by moving a chain of one or two consecutive nodes to a different location until no further improvement can be obtained. An example of the *Or-opt* exchange is given in Fig. 5.

(D) *Repair strategy*: Sometimes, the crossover operation inevitably violates some constraints and generates non-feasible solutions. Thus a repair strategy is presented to maintain a feasible solution pool. The repair strategy is described as follows.

Algorithm 3 (*Pseudo-code for repair strategy*)
Step 1: Sequence the selected coils in the current solution according to descending width and ascending due-date.
Step 2: If the groove constraint is satisfied, go to *Step* 3;
Else, delete a coil with a specific width violating the groove constraint, randomly select a new coil with different width from manufacturing orders and insert it to the current rolling round with the least cost, and go to *Step* 2.
Step 3: If the short slab constraint is satisfied, go to *Step* 4;
Else, delete a short slab from the rolling round, select a non-short slab from manufacturing orders randomly and insert it to the current sequence with the least cost without violating the groove constraint, and then go to *Step* 3.
Step 4: Stop this repair strategy, and return the scheduling solution.

After some criteria are satisfied, the MGA terminates the iterative process and reports the best schedule solution so far. The termination criteria can be a certain number of generations (*Gen*) or a given computation time. In the study, the algorithms are coded in C++, and compile with *MS Visual Studio 6.0*. The initial parameters are set as follows: the population size $Pop=200$, the selection pressure $c=0.1$, the similarity coefficient threshold $c_0=0.5$, the crossover probability $p_c=0.95$, and the mutation probability $p_m=0.05$. By simulating a set of production data collected from a hot rolling mill, the convergence curve of the proposed MGA is shown in Fig. 6.

One can see that the proposed MGA can converge to a satisfactory solution within a few hundreds of generations.

4.2.2. Local improving algorithm: τ-extremal optimization

In the HSM scheduling problem, the local evaluation function plays an important role in evaluating a scheduling solution. In this section, a novel local improving algorithm—τ-EO is presented to exploit the quality of a specific scheduling solution.

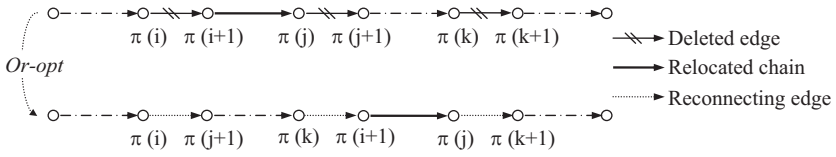

Fig. 5. Example of the *Or-opt* exchange: moving chain $(\pi(i+1), \pi(j))$ to the position between $\pi(k)$ and $\pi(k+1)$.

4.2.2.1. Introduction to extremal optimization. Extremal optimization (EO) is inspired by the self-organized critical models in ecosystems [24]. For a sub-optimal solution, the EO algorithm eliminates the components with extremely undesirable performance and then replaces them with randomly selected new components. Finally, a better solution may be generated by repeating such kinds of local search process. For a general minimization problem, the basic EO algorithm proceeds as follows:

Algorithm 4 (*Pseudo-code for basic τ-EO algorithm*)
Step 1: Initialize a solution S, and set $S_{best}=S$
Step 2: For the current solution S
(a) evaluate local fitness λ_i for each variable x_i,
(b) find j with $\lambda_j \geq \lambda_i$ for all i, i.e., x_j has the worst local fitness,
(c) choose at random $S' \in N(S)$ such that the "worst" x_j change its state,
(d) if the optimization objective $F(S') < F(S_{best})$, then set $S_{best} \leftarrow S'$,
(e) accept $S \leftarrow S'$ unconditionally, independently of $F(S')-F(S_{best})$.
Step 3: Repeat *Step* 2 as long as desired.
Step 4: Return S_{best} and $F(S_{best})$.

It is obvious that the basic EO algorithm has no parameters, which can be adjusted for selecting better solutions. To improve its optimization performance and avoid the possible dead ends, a general modification of the EO algorithm called τ-EO is proposed by introducing an adjustable parameter [25]. In the τ-EO algorithm, all variables x_i are ranked according to their fitness values λ_i, namely, find a permutation Π: $\lambda_{\Pi(1)} \geq \lambda_{\Pi(2)} \geq \cdots \geq \lambda_{\Pi(n)}$. Subsequently, each variable x_i to be updated is selected according to a probability distribution $P_k \propto k^{-\tau}$, $1 \leq k \leq n$, where k is the rank of the variable x_i. The power law distribution ensures that no ranks get excluded for further evolution while maintaining a bias against variables with bad fitness [25]. In the past decade, the EO algorithm and its derivatives have been extensively applied to solve various combinatorial optimization problems. Simulation results proved that the EO algorithm outperforms other state-of-the-art algorithms in many applications, such as graph bi-partitioning, satisfiability (MAX-K-SAT), TSP problems and some industrial applications [24–27].

4.2.2.2. Extremal optimization for improving the body section. Since the local transition costs between adjacent coils are the main parts of the scheduling optimization objective, the local search τ-EO algorithm is applied to improve the HSM scheduling solution. The local fitness is defined as $\lambda_i = c_{p(i),i} + c_{i,s(i)}$, where $p(i)$ and $s(i)$ denotes the predecessor and the successor of coil i, respectively. It means that the local fitness of a scheduled coil i in a rolling round is the sum of two sequence-dependent penalties which can be calculated by using only local information.

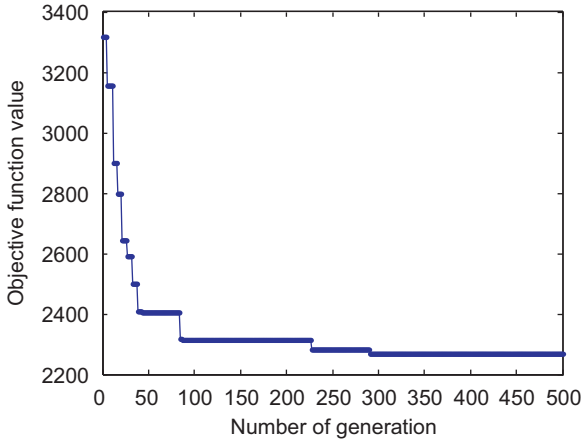

Fig. 6. Convergence curve of the modified genetic algorithm.

Algorithm 5 (*Pseudo-code for local improving algorithm—τ-EO*)
Step 1: Initialize parameters and obtain an initial solution S, which can be inherited from other algorithms, and then calculate the optimization objective $F(S)$, set $S_{best}=S$.
Step 2: For the current solution S, evaluate the local fitness λ_i for all scheduled coils and rank them according to their fitness values.
Step 3: Select a coil $c(s)$ according to the power law distribution $p_k(\tau_0)$, where τ_0 is a given parameter value.
Step 4: Choose the best solution S' from a neighboring subspace $N(S)$ of the current solution S.
Step 5: If $F(S') < F(S_{best})$, then set $S_{best} \leftarrow S'$.
Step 6: Accept $S \leftarrow S'$ unconditionally.
Step 7: If termination criteria are not satisfied, go to *Step* 2; else go next.
Step 8: Return S_{best} and $F(S_{best})$.

Note that in *Step 4* the subspace $N(S)$ can be generated through various strategies. In this paper, the "route-improvement" algorithm which is similar to the perturbation moves [16] is presented to generate the neighboring subspace. We take a scheduling solution S and improve it by slight perturbations. The perturbation is iterated until no further improvement is possible, and then the local optimum S' is obtained. The perturbation processes can be described as follows:

Delete a selected coil $c(s)$ from the current rolling sequence S;
Select an unscheduled coil $c(u)$, and insert it into the least-cost position;
Accept the solution S' as an element of $N(S)$ if the new rolling sequence is feasible and the local fitness $\lambda_{c(u)} < \lambda_{c(s)}$;
Repeat the above steps until all unscheduled orders have been processed.

The above perturbation moves are usually employed by human schedulers. Therefore, the local optimum in the neighboring subspace is intuitively reasonable in real scheduling systems.

Because the τ-EO algorithm has only one parameter τ, its optimal choice is critical for improving the optimization performance. A number of experimental and theoretical research efforts have been devoted to analyzing the optimal parameter selection for different combinatorial optimization problems [27,28]. In the HSM scheduling problem, a set of simulations indicate that the algorithm reaches the best solution with high probability at a prediction value $\tau_{opt} \approx 2.0$, and the objective function value seems to rise gradually around this τ_{opt} value for different scheduling instances [5]. Using the solution generated by the previous MGA as the initial solution, the τ-EO algorithm has the convergence curve as shown in Fig. 7.

It is obvious that the algorithm can improve the initial solution significantly and converges to a solution within 2000 generations. Note that the optimization process takes less than 45 s.

4.2.2.3. Hybrid evolutionary algorithms. The hybrid evolutionary algorithms combine the MGA and the τ-EO algorithms in different ways. First of all, the best scheduling solution generated by the MGA is further optimized by the τ-EO algorithm. This scheme is quite straightforward but shows a weak integration. Secondly, multiple solutions of the MGA are improved by the τ-EO algorithm for increasing the diversity of initial solutions. Thirdly, an

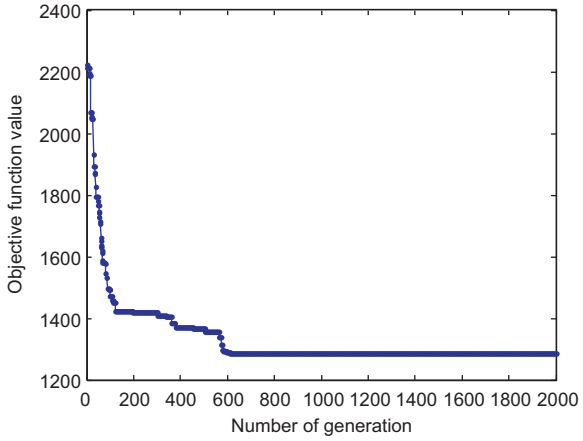

Fig. 7. Convergence curve of the τ-EO algorithm.

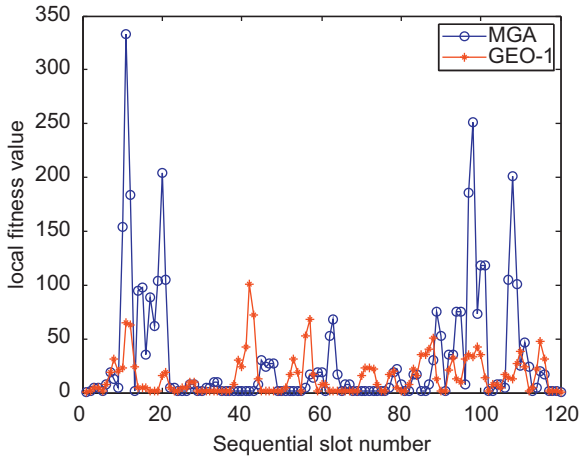

Fig. 8. Local fitness comparison between the MGA and the GEO-1.

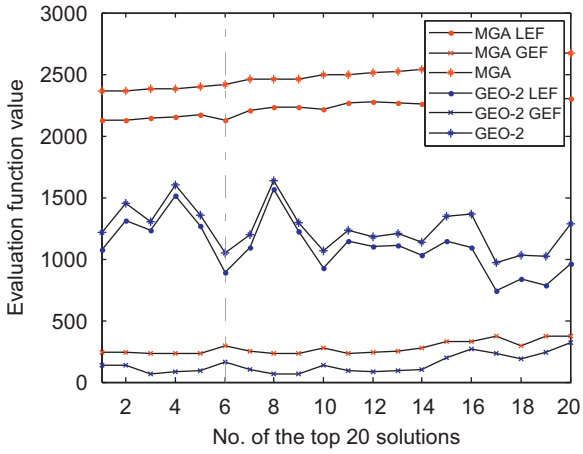

Fig. 9. Fitness comparison of multi-solution generated by the MGA and the GEO-2.

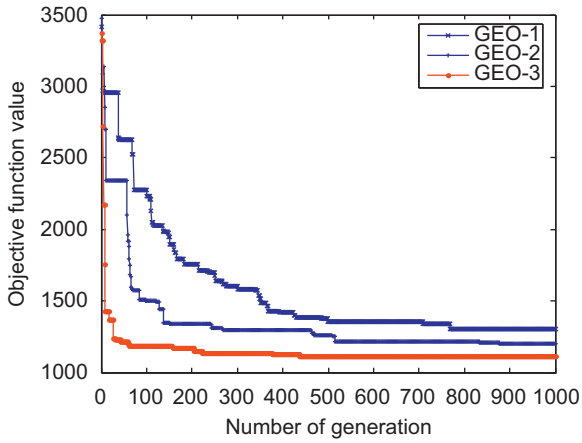

Fig. 10. Comparison on the Evolutionary processes of different combination schemes.

integrated method is presented, in which the τ-EO algorithm acts as the mutation operator of the MGA. Simulation results show that the third combination scheme provides better results than other schemes. Below we summarize and compare the simulation results of the above three hybrid evolutionary methods: GEO-1, GEO-2 and GEO-3.

(i) *GEO*-1 (*the best solution of the MGA is optimized by the τ-EO*): Fig. 8 compares the local fitness (i.e., local sequence-dependent transition costs) of the best solution of the MGA with that of the final solution of the GEO-1.
It is obvious that the GEO-1 can generate a scheduling solution with much lower sequence-dependent transition costs by iteratively replacing the undesirable or underperformed coils in a rolling sequence.

(ii) *GEO*-2 (*top* 20 *solutions of the MGA are further optimized by the τ-EO*): In the GEO-2, the top 20 solutions in the final population of the MGA are improved by the τ-EO algorithm. One can see from Fig. 9 that the GEO-2 can considerably improve both local fitness (i.e., the LEF of evaluating the sequence-dependent transition costs and the non-execution penalties) and global fitness (i.e., the GEF of evaluating the earliness/tardiness penalties). It is worth noting that in the above figure the multiple solutions of the MGA are ordered according to the value of the optimization objective $F(S)$. Obviously, an initial solution with a "worse" fitness value may be improved to a "better" scheduling solution by the GEO-2, such as the no. 6 solution of the top 20 solutions in Fig. 9.

(iii) *GEO*-3 (*using τ-EO as mutation operations in MGA*): In the GEO-3, the τ-EO algorithm is used as the mutation operator of the MGA. So, a number of $Pop \times p_m$ solutions are optimized through the τ-EO algorithm in each generation of the MGA. Using the same initial solution, the evolutionary processes of the GEO-1, the GEO-2 and the GEO-3 are illustrated in Fig. 10.

In the simulation of a wide range of industrial data, the GEO-3 almost always provides the best scheduling solution. Therefore, the GEO-3 is selected as the optimization engine of our HSM scheduling system.

4.3. Design of a HSM scheduling system

This scheduling system is defined as a configurable tool that solves the HSM scheduling problem to maximize production throughput, improve product quality and customer service (i.e., on-time delivery) levels. The schematic structure of the scheduling prototype system is presented in Fig. 11.

The HSM scheduling system first downloads manufacturing orders from the upper-level ERP systems, and then generate computer-aided scheduling scenarios for single rolling round or multiple rounds (i.e., a campaign) by the hybrid evolutionary algorithm as discussed in Section 4.2. The system assumes that

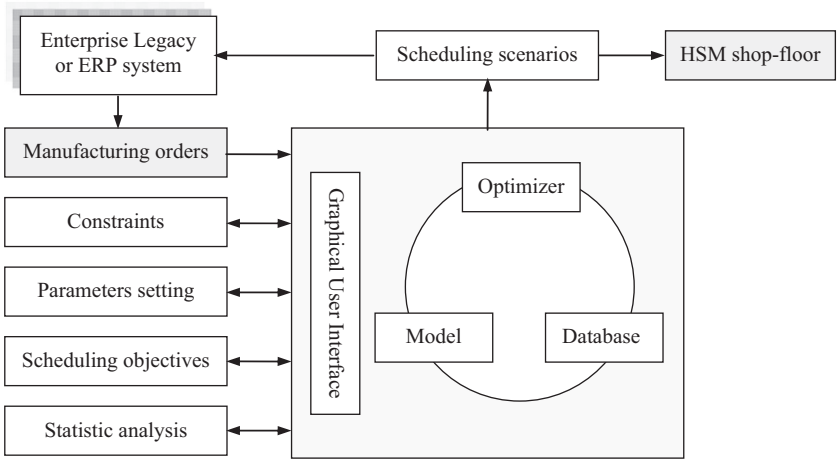

Fig. 11. Schematic structure of the HSM scheduling system.

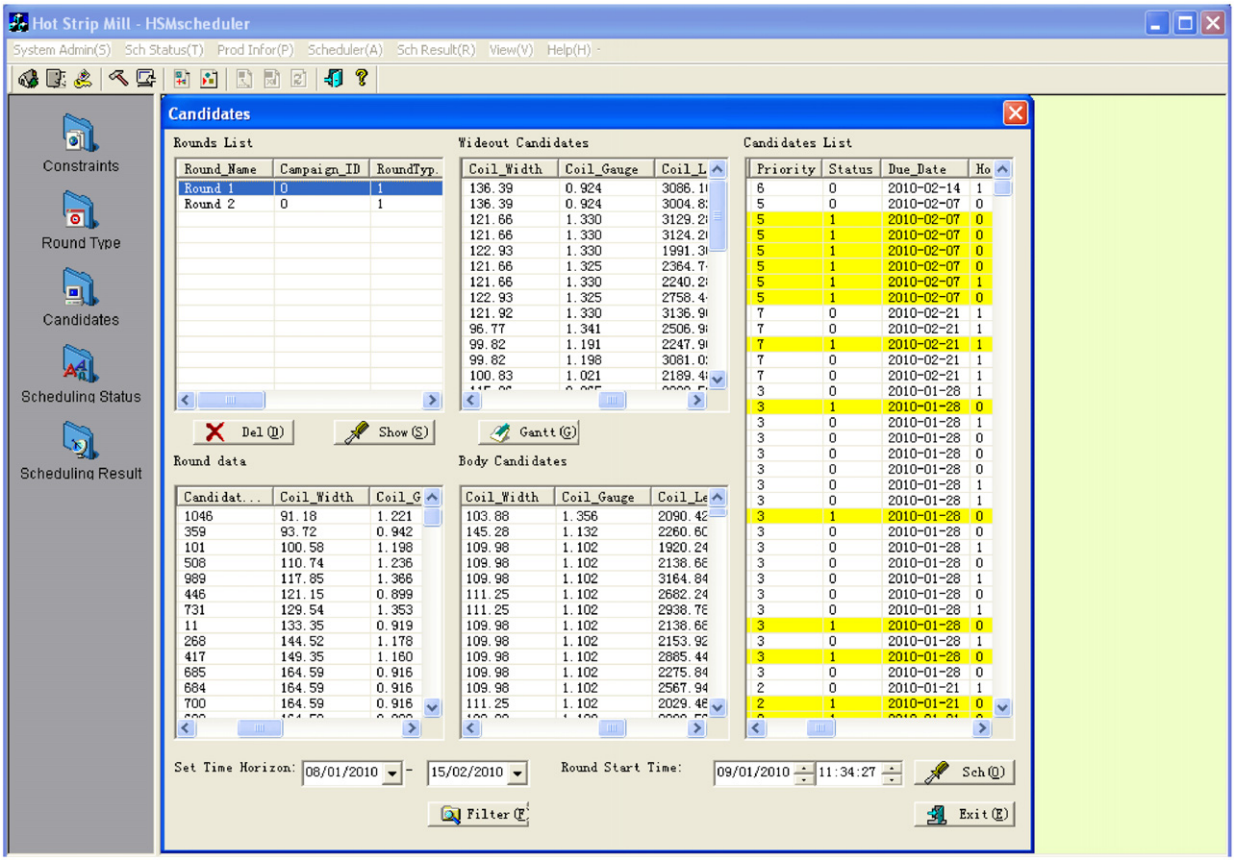

Fig. 12. Main Graphical User Interface of the developed HSM scheduling system.

human planners are allowed to guide the scheduling process and tune the optimization parameters for imposing human decision-making. The semi-automatic scheduling process is described as follows:

Step 1: The human planner sets the scheduling time horizon to select available slabs according to the corresponding coils in manufacturing orders. The scheduling system allows planners to adjust the selection rules based on the attributes of manufacturing orders (e.g., due-date, priority);

Step 2: Select the type of rolling rounds (i.e., coffin type selection). In order to help the planner make a reasonable choice, the system provides detailed information of different round types;

Step 3: Based on the site-specific production requirements, the planner can configure the constraints settings;

Step 4: Under the pre-configured conditions, the planner starts the optimization engine to generate scheduling solutions;

Step 5: Evaluate the generated scheduling solution. The planner can check the scheduling solutions by viewing the graphical width (or gauge, hardness, etc.) patterns and the statistical analysis results. The planner can also edit a specific rolling round by using graphical user interfaces, such as, inserting, replacing,

moving or deleting slabs from a rolling round, or rescheduling some rolling rounds in the existing scheduling scenario.

Since the object-oriented software technology has gained wide recognition as a preferred approach for building and maintaining complex application programs, the scheduling system is developed under the object-oriented platform of *MS Visual Studio 6.0* and *MS SQL Server*. The main graphical user interface of the developed scheduling system is shown in Fig. 12.

In the developed HSM scheduling system, the human planner can view the detailed profiles of the scheduled rolling rounds in width, gauge, and hardness transitions and jumps through graphical user interfaces. He/she also can adjust the existing rolling rounds by inserting, replacing, moving or deleting scheduled slabs. In addition, the scheduling solutions can be uploaded to the upper-level management systems and downloaded to the shop-floor manufacturing execution system.

5. Computational results

In this section, we first consider a set of production data, which consists of 1050 manufacturing orders with 336 short slabs and 75 warm-up slabs. The specifications of the corresponding coils mainly includes width (range: 88.90–164.59 cm), gauge (range: 0.883–1.404 cm), hardness (range: 1–10), length and finished temperature (range: 950–1450 °C). The scheduling objective can be calculated on the basis of the penalty table and the mathematical model in Section 3. Fig. 13 shows the width, gauge and hardness transition patterns of a rolling round.

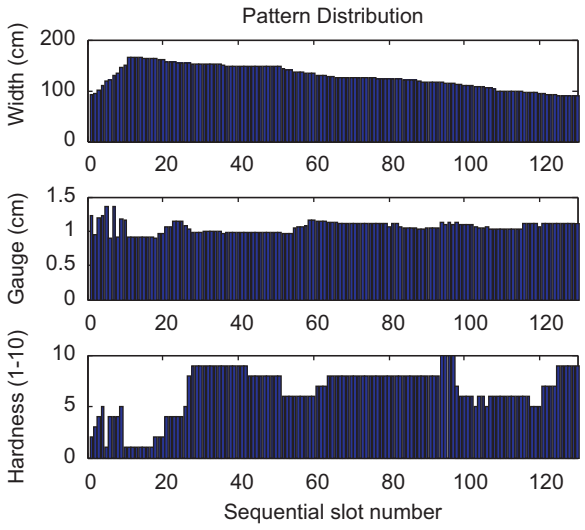

Fig. 13. Width, gauge and hardness transition patterns of a rolling round example.

One can see in Fig. 13 that the width profile of the rolling round follows a standard "coffin-shape" pattern, and the gauge and hardness transitions are smooth. It is worth noting that the scheduling of the rolling round only takes about 320 s on a Pentium 2.4 GHz CPU. The campaign can also be constructed by generating rounds one by one. Fig. 14 shows the width transition pattern of a rolling campaign. The optimization objective values of the five consecutive rolling rounds are 1112, 1733, 2639, 4107 and 3505, respectively.

During generating the rolling campaign, the optimization objectives for the MGA and the GEO-3 are reported in Table 2.

Note that the rolling rounds usually become worse, without updating the manufacturing order pool. To simulate a dynamic environment, we import 200 new coils into the set of manufacturing orders after the scheduling of each rolling round. The optimization objectives for the heuristic algorithm 2 in Section 4.2.1, the MGA and the GEO-3 are reported in Table 3.

It is obvious in Tables 2 and 3 that the proposed hybrid evolutionary algorithm can considerably improve the optimization objective. The HSM scheduling system equipped with the optimization engine can obtain an optimized rolling round within 600 s. Furthermore, extensive computational results on industrial data

Table 2
Improvement obtained by the proposed GEO-3 without updating manufacturing orders.

Round no.	MGA			GEO-3			Improvement (%)
	LEF value	GEF value	Objective	LEF value	GEF value	Objective	
1	1916	90	2006	1032	80	1112	44.57
2	2443	230	2673	1583	150	1733	35.17
3	3973	260	4233	2539	100	2639	37.66
4	4947	370	5317	4007	100	4107	22.76
5	8162	220	8382	3385	120	3505	58.18

Table 3
Improvement obtained by the proposed GEO-3 with updating manufacturing orders.

Instance no.	Heuristic algorithm 2	MGA	GEO-3	
	Objective	Objective	Objective	Improvement (% over MGA)
1	4129	2404	1523	36.65
2	3824	1715	1341	21.81
3	3958	2253	1641	27.16
4	4007	1971	1374	30.29
5	4203	2685	2030	24.39

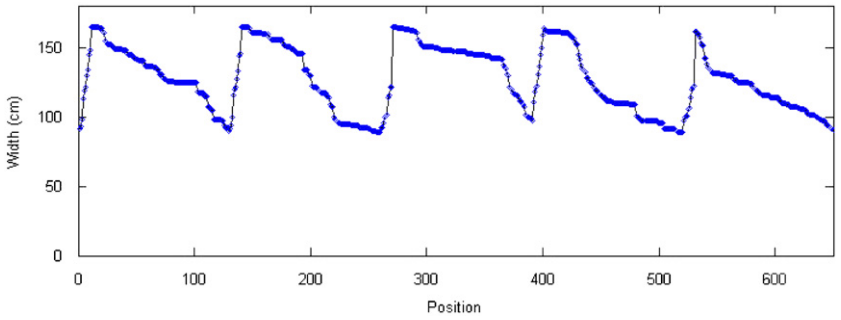

Fig. 14. Width transition pattern of a rolling campaign.

demonstrate that the developed HSM scheduling system has superior performances in scheduling quality and computing efficiency.

6. Conclusions

In this paper, we study the HSM scheduling problem in the steel industry. A mathematical model is formulated to describe two important scheduling sub-tasks: (1) selecting a subset of manufacturing orders; (2) generating an optimal rolling sequence. In view of the complexity of the scheduling problem, hybrid evolutionary algorithms are proposed through the combination of genetic algorithm (GA) and extremal optimization (EO). With the help of the developed HSM scheduling system, simulations are conducted to demonstrate that the hybrid evolutionary algorithms can generate optimized rolling rounds or campaign efficiently. Although the hybrid evolutionary algorithm in this paper is developed to solve the HSM scheduling problem, we believe that it has great potentials in the areas of scheduling and optimization. Our future work will emphasize the generalization of this hybrid evolutionary optimization method.

Acknowledgment

The authors would like to thank the anonymous reviewer for his insightful and helpful comments. Chen Y.W. is grateful to Dr. Michael Pryce, Dr. Mingjie Zhao and Mr. Dawei Tang with the University of Manchester for improving the writing of the paper. The research is partially supported by Honeywell Technology & Solutions Lab, Shanghai, and National Natural Science Foundation of China (no. 60574063, 61074150).

References

[1] Lee HS, Murthy SS, Haider SW, Morse DV. Primary production scheduling at steelmaking industries. IBM Journal of Research and Development 1996;40(2):231–52.
[2] Tang LX, Liu JY, Rong AY, Yang ZH. A review of planning and scheduling systems and methods for integrated steel production. European Journal of Operational Research 2001;133:1–20.
[3] Balas E. The prize collecting traveling salesman problem. Networks 1989;19:621–636.
[4] Lopez L, Carter MW, Gendreau M. The hot strip mill production scheduling problem: a tabu search approach. European Journal of Operational Research 1998;106:317–335.
[5] Chen YW, Lu YZ, Yang GK. Hybrid evolutionary algorithm with marriage of genetic algorithm and extremal optimization for production scheduling. International Journal of Advanced Manufacturing Technology 2008959–968.
[6] Kosiba ED, Wright JR, Cobbs AE. Discrete event sequencing as a traveling salesman problem. Computers in Industry 1992;19(3):317–327.
[7] Assaf I, Chen M, Katzberg J. Steel production schedule generation. International Journal of Production Research 1997;35(2):467–477.
[8] Tang LX, Wang XP. Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem. The International Journal of Advanced Manufacturing Technology 2005;12:1433–3015.
[9] Peterson CM, Sorensen KL, Vidal RVV. Inter-process synchronization in steel production. International Journal of Production Research 1992;30:1415–25.
[10] Yadollahpour MR, Bijari M, Kavosh S, Mahnam M. Guided local search algorithm for hot strip mill scheduling problem with considering hot charge rolling. International Journal of Advanced Manufacturing Technology 2009;45:1215–31.
[11] Verdejoa VV, Alarcób MAP, Sorlí MPL. Scheduling in a continuous galvanizing line. Computers & Operations Research 2009;36:280–296.
[12] Fang HL, Tsai CH. A genetic algorithm approach to hot strip mill rolling scheduling problems. In: Proceedings of the tenth IEEE international conference on tools with artificial intelligence, 1998. p. 264–71.
[13] Chen X, Wan W, Xu X. Modeling rolling batch planning as vehicle routing problem with time windows. Computers & Operations Research 1998;25(12):1127–36.
[14] Tang LX, Liu JY, Rong AY, Yang ZH. A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron & Steel Complex. European Journal of Operational Research 2000;124:267–82.
[15] Stauffer L, Liebling TM. Rolling horizon scheduling in a rolling-mill. Annals of Operations Research 1997;69:323–49.
[16] Cowling P. A flexible decision support system for steel hot rolling mill scheduling. Computers & Industrial Engineering 2003;45:307–321.
[17] Knoop P, Van Nerom L. Scheduling requirements for hot charge optimization in an integrated steel plant. In: Proceedings of the industry applications conference, 38th IAS annual meeting, 2003. p. 74–8.
[18] Okano H, Davenport AJ, Trumbo M, Reddy C, Yoda K, Amano M. Finishing Line Scheduling in the steel industry. IBM Journal of Research and Development 2004;48(5/6):811–30.
[19] Han J. Local evaluation functions and global evaluation functions for computational evolution, 2003. DOI: SFI-WP 03-09-048.
[20] Refael H, Mati S. Machine scheduling with earliness, tardiness and non-execution penalties. Computers & Operations Research 2005;32:683–705.
[21] Michalewicz Z. Genetic algorithms+data structures=evolution programs. Springer; 1996.
[22] Chang WA, Ramakrishna RS. A genetic algorithm for shortest path routing problem and the sizing of population. IEEE Transaction on Evolutionary Computation 2002;6(6):566–79.
[23] Larrañaga P, Kuijpers CMH, Murga RH, Inza I, Dizdarevic S. Genetic algorithms for the travelling salesman problem: a review of representations and operators. Artificial Intelligence Review 1999;13:129–70.
[24] Boettcher S, Percus AG. Extremal optimization: methods derived from co-evolution. In: Proceedings of the genetic and evolutionary computation conference, 1999. San Francisco: Morgan Kaufmann; 825–32.
[25] Boettcher S, Percus AG. Nature's way of optimizing. Artificial Intelligence 2000;119:275–86.
[26] Sousa FL de, Vlassov V, Ramos FM. Generalized extremal optimization: an application in heat pipe design. Applied Mathematical Modeling 2004;28:911–31.
[27] Chen YW, Lu YZ, Chen P. Optimization with extremal dynamics for the traveling salesman problem. Physica A 2007;385:115–23.
[28] Boettcher S, Percus AG. Optimization with extremal dynamics. Complexity 2003;8:57–62.
[29] Chen AL, Yang GK, Wu ZM. Production scheduling optimization algorithm for the hot rolling processes. International Journal of Production Research 2008;46(7):1955–73.