A CMA-ES algorithm for solving the steelmaking scheduling problem involving time buffers

Sheng-Long Jiang
School of materials science and engineering
Chongqing University
Chongqing, China
sh.l.jiang.jx@gmail.com

Wei Tang
School of materials science and engineering
Chongqing University
Chongqing, China
wei.tang@cqu.edu.cn

Wanzhe Hu
School of Economics and Management,
Chongqing University of Posts and Communications
Chongqing, China
wanzhe.hu@gmail.com

***Abstract*— The production system of a steelmaking plant always is thought of as a special hybrid flow shop (HFS). However, most scheduling models are not performed very well in practical environments due to practical problems are more complex than classic versions, and the algorithms are weak in solving problems with various scales and structures. This paper studies a practical scheduling problem involving time buffers, and propose an improved covariance matrix adaptation evolution strategy (CMA-ES) algorithm without painstaking parameter selections. First, the studied problem is formulated as a mixed-integer linear programming (MILP) model. Second, we develop an improved CMA-ES algorithm with a problem-oriented encoding and decoding scheme, and a differential evolution (DE) based restart policy. Finally, the proposed algorithm is compared with the standard DE algorithm and the state-of-the-art algorithm named CCABC. Experimental results carried out on a variety of synthetic scheduling benchmarks demonstrate that the proposed technique shows the advantage in seeking optimums.**

Keywords—scheduling, buffer, CMA-ES, hybrid flow shop, steelmaking

I. INTRODUCTION

The steelmaking plant is a critical building block in a modern iron & steel manufacturing company, which mainly consists of a steelmaking stage, a single or multiple refining stages, and a casting stage (as shown in Fig. 1). The mission of a steelmaking plant is to produce qualified slabs or billets with suitable chemical percentages and physical shapes. In a practical industrial system, each job, molten steel poured from blast furnaces, must travel in a specific channel across from the steelmaking to the casting stage but may skip some refining stages. During the flow sheet, molten steel is poured into an intermediate vessel named ladle, and transfer to a continuous casting machine for solidification, in which a number of jobs are grouped into in a batch and processed without any time stopped due to its key part named Tundish is only available within a specific time range. Because of the requirement on high temperature (1600℃ above) and the strict chemical spectrums, each job must provide some time buffer to prepare processing materials and tools, and then move to the next stage. Concerning these topological flow and technical requirements, the steelmaking scheduling problem is commonly identified as a special hybrid flow shop (HFS) scheduling problem with complex constraints[1][2], including setup, continuity, and transferring, etc. After the job sequence and machine allocation are determined, a feasible schedule implemented in a steelmaking plant can be illustrated with a Gantt diagram containing serval continuous blocks in the last stage (as shown in Fig. 2).

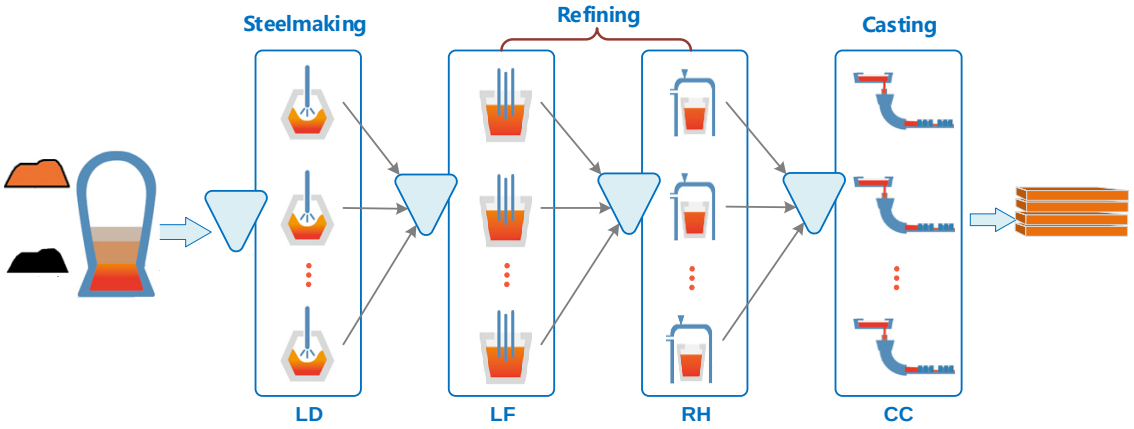

Fig. 1. Flowsheet of a steelmaking plant

This work is supported by the National Natural Science Foundation of China (No. 6187020702), the Opening Fund of Guangxi Key Laboratory of Automatic Detection Technology and Instrument (No. YQ19202), the China Scholarship Council.

In recent years, the steelmaking scheduling problem is extensively studied both in academia and industry, because of its strong industrial background. However, there is very little

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

literature on its real-world application. The main reason is that most of the steelmaking scheduling models are simply assumed to be variants of the classic HFS scheduling problem, and cause a wide variety of related solving algorithms to be Utopian theories. In this study, we address these issues by modeling the steelmaking scheduling problems that considers more practical constraints and developing a high-efficiency solving algorithm. Section II reviews related works about steelmaking scheduling problems and a powerful optimization algorithm named covariance matrix adaptation evolution strategy (CMA-ES). Section III provides a mathematical model formulating the practical scheduling problem. In Section IV, we develop an improved CMA-ES algorithm integrating prior knowledge towards the steelmaking scheduling problem. In Section V, empirical studies are carried out to verify the effectiveness of the proposed algorithm. The final section draws some conclusions of this study and points out some future study topics.

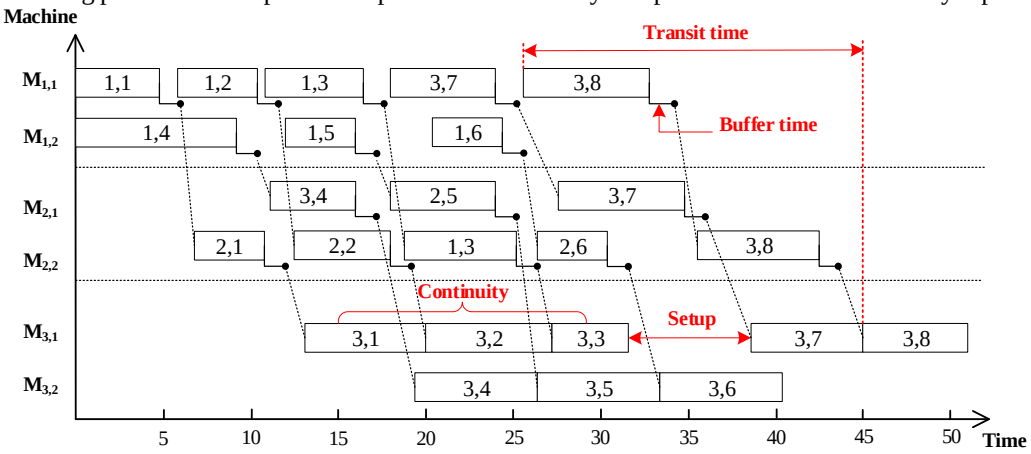

Fig. 2. Gantt diagram for a feasible schedule executed in a steelmaking plant

II. Literature Review

From the 1990s, researchers have developed a large variety of models to describe the scheduling problem in steelmaking plants, most of which are assumed to be variants of a classic HFS scheduling problem. Tang et al. [1]-[4] proposed mathematic models involving batching, precedence, setup and transferring constraints, and punctual delivery, job waiting and make-span objectives. However, due to the complexity of the practical industrial system, the released schedule grounded on these models often suffers from performance variation and constraint violations. Therefore, a wide range of dynamic [5]-[9] and uncertain [10]-[14] models were also reported in recent publications. The main reason caused the instability and infeasibility is that most models are too Utopian to be implemented in a practical production system. Therefore, many practitioners try to seek reasonable models taking account of additional practical constraints. Tang et al. [15] integrated transporter-related constraints and objectives into the steelmaking scheduling model. Missbauer et al. [16] proposed a realistic scheduling model with controllable casting speeds and balanceable production capacities. Tan and Liu [17] developed a scheduling model considering a variable electricity price. Xu et al. [18] studied a batching-related scheduling problem in which allocating jobs in batches needed to be determined. Amid these practical constraints, the time buffer is one of the most important factors, because it is not only used for further processing molten steel but also provide flexibility to resist some uncertain disturbances.

To solve these various scheduling problems, a wide range of problem-specific optimization algorithms have been raised. These approaches are mainly categorized into three types: (1) Heuristics based on dispatching rules [16]. (2) Mathematic programming mainly includes decomposition-based MILP [19] and Lagrangian relaxation [20]. (3) Evolutionary algorithm (EA), such as ant colony optimization (ACO) [4], differential evolution (DE) [5][23], artificial bee colony (ABC) [21][22], fruit fly optimization (FFO) [7] [24], etc. Among these algorithms, the EA is known as one of the powerful optimization tools for solving scheduling problems, because it scales well to high-dimension problems, and is robust to complex optimization problems. When applying an EA to solve steelmaking scheduling problems, we need to simplify a candidate solution as a genome and evaluate its objective value or fitness. For example, Pan et al. [21] and Li et al. [23][24] defined a job sequence at the first stage to represent as the scheduling genome, Tang et al. [5] defined a matrix with job sequence and machine allocation in all stages. The first way is a partly encoding way and is high efficient in the evolutionary process, but it is difficult to obtain an optimal evaluation in the sub-space represented by the vector gene. The second way is a complete encoding way and is well in seeking a local optimization evaluation, but it is weak in evolutionary optimization. Both of them are common for encoding HFS scheduling problem, and only modify the evaluation method in line with the problem characteristics. Therefore, some studies try to define problem-oriented scheduling genes. Jiang et al. [23] proposed a representation with a job sequence in the last stage, and develop an iterative backward list scheduling procedure. Pan [22] devised a special job sequence expression in which each job index was replaced with its batch index.

Another shortage of applying EAs in practice is that

control parameter tuning is indispensable before running an algorithm [25]. It cannot adapt very well to the practical scheduling problem with various scales and structures. In recent years, the CMA-ES is thought of as a top-notch black-box optimization tool towards non-convex, multimodal, discontinuous, or ill-conditioned problems in continuous spaces [26][27]. It applies a covariance matrix to learn autonomously parameter distributions and dependencies, and to generate new solutions. It is also regarded as a variant of estimation of distribution algorithm (EDA) with Gaussian distribution. CMA-ES provides an extremely prospective in solving many machine learning tasks [28][29], because it has no discrepancy between the behaviors toward the varied nature of black-box problems and is easily generalizable. Due to these advantages, many improved variants of CMA-ES came out [30]-[33].

As claimed by the tendency of modeling and the shortages of algorithms, we attempt to construct a practical scheduling model considering additional constraints and devise a CMA-ES algorithm to solve it.

III. Problem Statement

In this section, we formulate the studied scheduling problem in the steelmaking plant with a mixed-integer linear programming (MILP) model.

A. Symbols

In this subsection, we list symbols used for illustrating the input and output information of the MILP model.

Indices:

b index of batches.
i index of stages.
j index of jobs.
k index of machines.

Sets:

I stage set, $I = \{1, \cdots i, \cdots, g\}$, and $I^U = I \backslash \{g\}$.
J job set, $J = \{1, \cdots j, \cdots, n\}$, and J_i denotes a job subset that needs to visit stage i.
B batch set, $B = \{1, \cdots, b, \cdots, N\}$.
B_b job set of the b^{th} batch with the size of N_b, and $B_{b,r}$ denotes the r^{th} job in batch b.
M_i machine set of stage i with the size of m_i, and $M_{i,k}$ denotes the k^{th} machine in stage i.
H_j stage list of job j with the size of h_j, and $H_{j,q}$ denotes the q^{th} stage visited by job j.

Parameters:

at_j arrival time of the feeding materials for job j.
$PT_{i,j}$ standard duration of task $\langle i, j \rangle$.
$TT_{i,i+1}$ transfer time from stage i to stage $i+1$.
Q_i predefined buffer time after stage i.
ut_b setup time of the b^{th} batch.
γ_1 objective coefficient of make-span.
γ_2 objective coefficient of transit times.
L sufficiently large positive constant.

Decision Variables:

$x_{i,j,k}$ binary variable, if task $\langle i, j \rangle$ is allocated on machine $M_{i,k}$, $x_{i,j,k} = 1$; otherwise $x_{i,j,k} = 0$.
y_{i,j_1,j_2} binary variable, if task $\langle i, j_1 \rangle$ and $\langle i, j_2 \rangle$ are contiguously processed, $y_{i,j_1,j_2} = 1$; otherwise $y_{i,j_1,j_2} = 0$.
$u_{k,b}$ binary variable, if batch b is allocated on machine $M_{g,k}$, $u_{k,b} = 1$; otherwise, $u_{k,b} = 0$.
z_{b_1,b_2} binary variable, if batch b_1 and b_2 are contiguously processed, $z_{b_1,b_2} = 1$; otherwise $z_{b_1,b_2} = 0$.
$S_{i,j}$ starting time of task $\langle i, j \rangle$.
$D_{i,j}$ departure time of task $\langle i, j \rangle$.

B. Objectives

In a practical steelmaking system, the mission of scheduling decision is to seek the optimal objectives involving both production efficiency and cost. The first one is defined by the maximum completion time (f_1), and the second one is defined by the total transit time (f_2).

$$f_1 = C_{max} = \max_{j \in J}\{D_{g,j}\} \tag{1}$$

$$f_2 = \sum_{j=1}^{n}\left(S_{g,j} - S_{1,j}\right) \tag{2}$$

The objective of the scheduling problem is formulated as,

$$(P) \qquad F = f_1 + f_2 \tag{3}$$

C. Constraints

1) Sequencing and allocation

$$\sum_{k=1}^{m_i} x_{i,j,k} = 1, \qquad \forall i \in I^U, j \in J_i \tag{4}$$

$$y_{i,j_1,j_2} + y_{i,j_2,j_2} = 1, \forall i \in I^U,\ j_1, j_2 \in J_i,\ j_1 \neq j_2 \tag{5}$$

$$\sum_{k=1}^{m_g} u_{k,b} = 1, \qquad \forall b \in B \tag{6}$$

$$\mathrm{z}_{b_1,b_2} + \mathrm{z}_{b_2,b_1} = 1, \forall b_1, b_2 \in B,\ b_1 \neq b_2 \tag{7}$$

Equation (4) and (5) ensure the uniqueness of the job allocation and sequencing, and (6) and (7) impose the same constraint on batches.

2) Timing constraints

$$S_{1,j} \geq at_j,\ \forall j \in J \tag{8}$$

$$S_{i_2,j} - D_{i_1,j} \geq TT_{i_1,i_2}, i_1 = H_{j,q}, i_2 = H_{j,q+1}, \quad \forall q \in \{1, .., h_j - 1\}, j \in J \tag{9}$$

$$D_{i,j} - S_{i,j} \leq PT_{i,j} + Q_i, \qquad \forall i \in I^U, j \in J \tag{10}$$

$$S_{i,j_2} - S_{i,j_1} + L\left(3 - x_{i,j_1,k} - x_{i,j_2,k} - y_{i,j_1,j_2}\right) \geq PT_{i,j_1},\ \forall i \in I^U, j_1, j_2 \in J, j_1 \neq j_2 \tag{11}$$

$$D_{i,j_1} - S_{i,j_2} + L\left(3 - x_{i,j_1,k} - x_{i,j_2,k} - y_{i,j_1,j_2}\right) \leq PT_{i,j_2},\ \forall i \in I^U, j_1, j_2 \in J, j_1 \neq j_2 \tag{12}$$

$$TPT_b = \sum_{r=1}^{N_b} PT_{g,B_{b,r}}, \qquad \forall b \in B \tag{13}$$

$$S_{g,B_{b2,1}} - S_{g,B_{b1,N_{b1}}} + L\left(3 - u_{k,b_1} - u_{k,b_2} - z_{b_1,b_2}\right) \geq TPT_{b_1} + ut_{b_2}, \forall b_1, b_2 \in B,\ b_1 \neq b_2 \tag{14}$$

$$S_{g,B_{b,r+1}} - S_{g,B_{b,r}} = PT_{g,B_{b,r}}, \quad \forall b \in B, r \in \{1,2,\dots,N_b - 1\} \tag{15}$$

Equation (8) denotes a job starts only after its feeding hot metal arrives, (9) means that a job starts in the current stage only after it departed from the previous stage and has been arrived at the stage, (10) indicates a not started task must be protected by a predefined time buffer, (11) means a machine is only able to processed one job at a time, (12) means a job must be moved from the buffer when the previous one on the same machine has completed, (13) defines the total duration of batch b, (14) means a machine is only able to process one batch at a time in the last stage, (15) expresses the continuity relationship between jobs in the same batch.

IV. Proposed Algorithm

A. Main idea of implementing CMA-ES

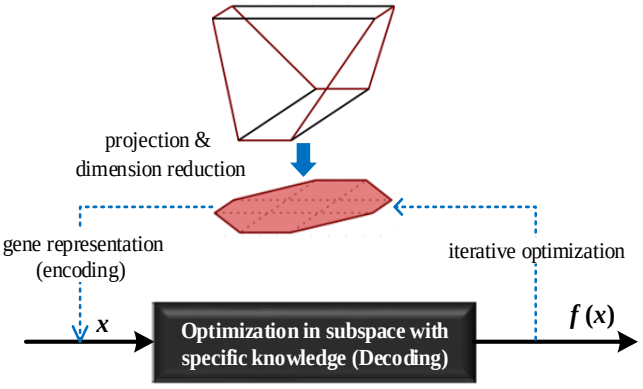

Fig. 3. Black-box optimization for scheduling

Comparing with the classic HFS scheduling problem, the search space of a steelmaking scheduling problem is not very huge, because the job sequence in each batch is deterministic. Therefore, we can represent the search space with a lower-dimensional genome (vector) and the primal problem can be simplified as an easy-to-solve subproblem. The optimization process is assumed to be a black box (as shown in Fig. 3). Since there is no available gradient information during the optimization process, we map the steelmaking scheduling problem with a real-valued vector and try to seek an optimal genome by the CMA-ES algorithm (as shown in Algorithm I). Moreover, the MILP problem stated in Section III is transformed into a continuous optimization problem, and supposed to be solved more quickly than mathematical solvers, like CPLEX, GUROBI, etc. In this optimization procedure, the most important issue is how to integrate the problem characteristics with EMA-ES to avoid a completely blind search. In the following sub-sections, we implement a novel encoding scheme, a high efficient evaluation method, a boundary handling, and a restart policy.

Algorithm I: CMA-ES

Step 0: initialize λ, m, σ, $C = I$, $p_C = 0$, $p_\sigma = 0$, NT
Step 1: while not terminated
1.1 Sample λ new solutions
for $k = 1$ to λ
(a) $\pi_k = sample\_new\_solution(m, \sigma^2 C)$ /* *ref.* [27]*/
(b) $F_k = fitness(\pi_k)$ /* *ref.* Section IV.C */
1.2 Selection and recombination
(a) $\pi_{1\cdots\mu} = select\_\mu\_best(\{(\pi_k, F_k) | 1 \leq k \leq \lambda\})$
(b) $m = update\_m(\pi_{1\cdots\mu})$. /**ref.* [27] */
1.3 Step-size control
(a) $p_\sigma = update\_psigma()$/**ref.* [27] */
(b) $\sigma = update\_sigma()$ /**ref.* [27] */
1.4 Covariance matrix adaptation
(a) $p_C = update\_pC()$/**ref.* [27] */
(b) $C = update\_C()$/**ref.* [27] */
1.5 Restart
If F_1 is not improved and reaches NT attempts then
$\pi_{1\cdots\lambda} = genrate\_new\ \_slutions()$ /* *ref.* Section IV.E */
return π_1

B. Real-valued encoding scheme (need to review again)

To make a good trade-off between external and internal optimization within the black box, we develop a novel real-valued encoding scheme incorporated with prior knowledge. Usually, the solutions of a classic HFS scheduling problem are represented as a job sequence in the first stage [21], last stage [23], or bottleneck stage [34]. Due to the continuity and setup constraints in the last stage, we need to seek an encoding vector for mapping the steelmaking scheduling problem which not only includes the sequence information but also embodies timing information.

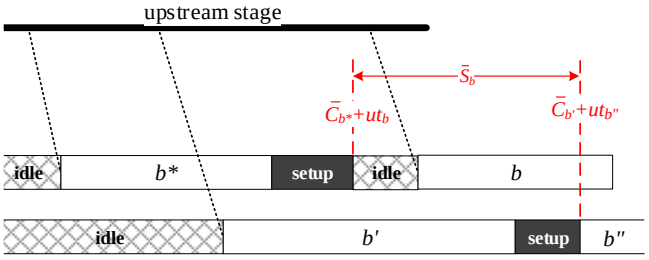

Fig. 4. Complete schedule in the casting stage

It is worth noted that a complete schedule in the last stage with parallel machines is composed of working and idle intervals (as shown in Fig 4). Once the processing sequence is fixed, the starting time of the current batch ($\bar{S}_b$), can be defined as $\bar{S}_b = S_{g,B_{b,1}} = \bar{C}_{b^*} + ut_b + \Delta_b$, where $\bar{C}_{b^*}$ is the completion time of the previous batch b' on the same machine as batch b. Additionally, we can also infer that the upper bound of $\bar{S}_b$, $\bar{S}_b \leq \bar{C}_{b'} + ut_{b''}$, where b' is the previous one of batch b in the given cast sequence. Then, we define $\Delta_b = \beta_b \times (\bar{C}_{b'} + ut_{b''} - \bar{C}_{b^*} - ut_b)$, where $\beta_b \in [0,1]$ is the idle ratio of batch b. In this paper, we identify the setup times of all batches are fixed the same value, then we can calculate

$\Delta_b = \beta_b \times (\bar{C}_{b'} - \bar{C}_{b^*})$.

Taking these cues, we devise a two-part real-value vector $\pi = (\alpha|\beta)$ to map the search space, where $\alpha \in [0,1]$ denotes the priority of each batch, and $\beta \in [0,1]$ denotes the idle ratio before it starts. Fig. 5 exemplifies the method, where five batches are encoded with a priority vector $\alpha = \{0.9,0.8,0.3, 0.5,06\}$ and a idle ratio vector $\beta = \{0.1,0.5,0.3,0.2,0.8\}$. According to this encoding scheme, we sort jobs in descending order of α and allocate machines following the earliest available machine (EAM) rule. Then, the final schedule in the last stage can be determined by insert fixed idle times indicated by β.

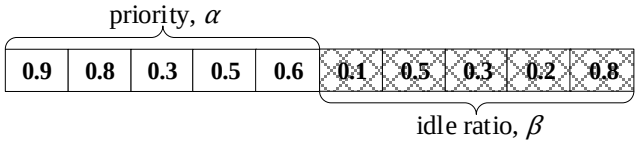

Fig. 5. Two-part encoding scheme

According to this encoding scheduling, we can determine a unique solution in the casting stage. It means that all decision variables are projected on the variables in the casting stage.

C. LP-based decoding Procedure

When decision variables are determined by a real-valued vector, we can apply a backward list scheduling to get a multi-dimensional vector with job list $X_{i,k}$ $(i < g)$ and batch list $U_{g,k}$ allocated on each machine. Then, all sequencing and allocation constraints of P can be removed, and timing constraint (11), (12) and (14) can be simplified as:

$$S_{i,j_2} - S_{i,j_1} \geq PT_{i,j_1},\ \forall i \in I^U, \langle j_1, j_2 \rangle \in X_{i,k} \quad (16)$$

$$D_{i,j_1} - S_{i,j_2} \leq PT^t_{i,j_2},\ \forall i \in I^U, \langle j_1, j_2 \rangle \in X_{i,k} \quad (17)$$

$$S_{g,B_{b_2,1}} - S_{g,B_{b_1,N_{b_1}}} \geq TPT_{b_1} + su_{b_2}, \langle b_1, b_2 \rangle \in U_{g,k} \quad (18)$$

where $\langle j_1, j_2 \rangle$ and $\langle b_1, b_2 \rangle$ are two consecutive jobs and batches on the same machine.

When sequencing and allocation variables are fixed, we formulate the following linear programming (LP) model which can be solved very easily by commercial solvers.

$$\text{LP: } \begin{cases} F = f_1 + f_2 \\ s.t. (8-10), (13), (16-18) \end{cases} \quad (19)$$

D. Boundary handling method

It is important to note that a new solution may violate its bound constraint [0,1] of the two-part-vector representation during the random sampling process. A good constraint handling method is able to restrict the feasible region and guide the algorithm toward a promising region [35]. In this paper, we apply a penalty function proposed by Igel et al. [31] to handle the boundary constraints.

$$F = F\big(feasible(\pi)\big) + 10^4 \|\pi - feasible(\pi)\|_1 \quad (20)$$

where

$$feasible(\pi) = \begin{cases} 0 & \pi_r < 0 \\ \pi_r & 0 \leq \pi_r \leq 1 \\ 1 & \pi_r > 1 \end{cases} \quad (21)$$

E. DE-based Restart policy

In some cases, a rugged landscape of the studied scheduling problem may cause the CMA-ES algorithm to explore an unpromising region for many trials. Therefore, the result is more likely to link with the initial population. It is necessary to restart the algorithm to search for another promising region. In this paper, we introduce a restart policy imitated the operator in DE to avoid trapping into local optimums after $NS/10$ attempts, which is detailed in Algorithm II.

Algorithm II: DE-based restart policy

for $r = 1$ to λ

(a) randomly choose two indices r_1, r_2 from $1 \ldots \lambda$, and $r \neq r_1 \neq r_2$

(b) $\pi_r = \begin{cases} \pi_r + \mathcal{N}(0,1) * (\pi_{r_1} - \pi_{r_2}), & if\ rnd < 0.5 \\ \pi_r & , otherwise \end{cases}$.

(c) $F_i = fitness(\pi_r)$ // optimized with a Solver.

end for

V. Computational experiments

Because the formulation and benchmarks of the studied scheduling problem have not been reported elsewhere, we conduct computational experiments in this section to analyze the performance of the proposed CMA-ES by comparisons. All the related algorithms are coded in C++ language, compiled with Visual Studio 2015, and executed on a win10 64bit Personal Computer with an 1.8550 GHz Intel Core i7-8550U CPU and 16.0GB RAM.

A. Instance generation

We generated test instances in a random way, in which the input data is collected from a large iron & steel company located in the north of China. The production system has more than one refining stage and is characterized as follows.

(1) The number of stages is four, and the number of machines in each stage is {3, 4, 3, 3}.

(2) The processing time $PT_{i,j}$ is generated with following uniform distributions, $U(\cdot,\cdot)$:

- At the steelmaking stage, $PT_{1,j} \sim U(27,32)$.
- At the first refining stage, $PT_{2,j} \sim U(40,60)$.
- At the second refining stage, $PT_{3,j} \sim U(40,60)$.
- At the casting stage, $PT_{4,j} \sim U(30,40)$.

(3) The transfer time $TT_{i,i+1}$ =6.

(4) The setup time $ut_b = 60$.

(5) The buffer time $Q_i = 5$.

In the following experiments, three levels of batch size $N \in \{6,9,12\}$ is considered, and the job size of each batch is a uniform random distribution, $N_b \sim U(8,12)$. We randomly generate seven instances of each batch level, thus a total of 21 scheduling benchmarks are synthesized.

B. Competing algorithms

To verify the effectiveness of the model as well as to compare the proposed CMA-ES of which initial parameters are recommended by [27], we have selected the standard DE algorithm and the CCABC as the competitors.

(1) DE, a well-known continuous EA, and its variants have successfully applied in solving a variety of steelmaking scheduling problems [5][13][18][23]. In this section, experiments, a standard DE (DE/rand/1/bin) algorithm is implemented with the same encoding and decoding scheme proposed in this paper. The control parameters are suggested by Ronkkonen et al. [36] where population size, $Np = 50$, scale factor, $F = 0.5$, crossover rate, $Cr = 0.9$.

(2) CCABC, a state-of-the-art EA for solving steelmaking scheduling problems which is similar to the studied problem. In the following experiments, we implemented the CCABC with its own encoding scheme but the decoding procedure proposed in this paper. The control parameters are recommended by Pan [22], where population size, $Np = 10$, neighbor search ratio, $\delta = 0.75$, and number of consecutive iterations, $\theta = 50$.

To make a fair competition, we randomly initialize the population of all algorithms and terminate them when the evaluation number reaches 5000. Especially, all decoding procedures based on MILP or LP models are solved by the commercial optimization solver GUROBI 9.0 (with default settings).

C. Experimental Results

For each testing instance, we independently conduct each algorithm 10 times to obtain the mean computational output. The relative percentage deviation (RPD), defined in (22) below, over the best solutions value found in the experiment is computed.

$$RPD = \frac{F - F_{\min}}{F_{\min}} \times 100\% \qquad (22)$$

where $F_{\min}$ is the minimum objective value obtained by running the same algorithm in a multiple repeating way.

TABLE I. COMPUTATIONAL RESULTS OF COMPARATIVE ALGORITHMS

No.	CMA-ES				DE				CCABC			
	min.(%)	*max.* (%)	*mean.* (%)	*T(s)*	*min.*(%)	*max.* (%)	*mean.* (%)	*T(s)*	*min.*(%)	*max.* (%)	*mean.* (%)	*T(s)*
1#	**0.000E+00**	3.030E+00	1.794E+00	36	2.403E+00	4.284E+00	3.448E+00	10	2.926E+00	1.996E+01	6.270E+00	14
2#	1.715E+00	4.609E+00	3.108E+00	41	**0.000E+00**	5.573E+00	3.537E+00	11	3.215E-01	1.115E+01	2.358E+00	16
3#	**0.000E+00**	2.564E+00	1.442E+00	39	2.350E+00	6.944E+00	6.197E+00	9	4.915E+00	1.603E+01	7.051E+00	14
4#	**0.000E+00**	4.019E+00	2.072E+00	42	2.056E+00	5.140E+00	3.458E+00	12	5.794E+00	1.804E+01	1.308E+01	18
5#	8.811E-01	5.727E+00	4.141E+00	37	**0.000E+00**	6.608E+00	3.084E+00	15	6.432E+00	2.282E+01	1.181E+01	19
6#	**0.000E+00**	4.102E+00	2.550E+00	38	4.435E+00	8.758E+00	7.095E+00	14	1.552E+00	3.492E+01	1.752E+01	20
7#	**0.000E+00**	4.958E+00	2.781E+00	40	2.297E+00	6.530E+00	3.748E+00	10	9.311E+00	3.301E+01	1.790E+01	18
8#	9.357E-01	4.971E+00	2.456E+00	55	**0.000E+00**	5.965E+00	2.924E+00	18	4.678E+00	1.591E+01	1.228E+01	27
9#	1.528E+00	5.501E+00	3.606E+00	53	2.078E+00	5.318E+00	4.034E+00	21	**0.000E+00**	3.368E+01	1.314E+01	30
10#	**0.000E+00**	4.715E+00	2.966E+00	51	7.072E+00	1.141E+01	8.897E+00	19	8.061E+00	1.954E+01	1.125E+01	26
11#	**0.000E+00**	5.139E+00	3.498E+00	53	4.568E+00	1.342E+01	7.780E+00	20	5.496E+00	3.269E+01	1.649E+01	26
12#	**0.000E+00**	3.041E+00	1.886E+00	57	2.433E-01	3.589E+00	3.041E-01	20	4.258E-01	1.052E+01	8.577E+00	27
13#	**0.000E+00**	3.351E+00	1.737E+00	52	4.617E+00	9.159E+00	7.074E+00	22	2.457E+00	2.278E+01	1.236E+01	30
14#	**0.000E+00**	2.717E+00	1.381E+00	54	4.705E+00	1.173E+01	7.886E+00	21	3.844E+00	2.180E+01	6.627E+00	29
15#	**0.000E+00**	2.137E+00	1.053E+00	68	**0.000E+00**	6.790E+00	4.463E+00	30	1.662E+00	2.284E+01	1.282E+01	40
16#	**0.000E+00**	3.065E+00	1.541E+00	73	9.045E-01	8.141E+00	6.131E+00	28	6.683E+00	1.930E+01	7.789E+00	38
17#	**0.000E+00**	2.572E+00	1.088E+00	69	4.797E+00	1.009E+01	7.962E+00	27	1.835E+01	2.977E+01	1.963E+01	37
18#	**0.000E+00**	2.642E+00	1.347E+00	75	1.677E+00	6.453E+00	4.370E+00	28	1.209E+01	3.547E+01	1.707E+01	37
19#	**0.000E+00**	1.406E+00	8.293E-01	70	4.867E-01	3.786E+00	1.947E+00	30	4.002E+00	2.466E+01	1.509E+01	42
20#	**0.000E+00**	2.991E+00	2.030E+00	71	3.752E+00	7.939E+00	5.982E+00	28	8.320E+00	3.018E+01	1.588E+01	38
21#	**0.000E+00**	3.233E+00	1.338E+00	72	5.422E+00	9.854E+00	8.238E+00	27	4.692E+00	2.419E+01	1.309E+01	37
avg.	2.409E-01	3.642E+00	2.126E+00	55	2.565E+00	7.499E+00	5.169E+00	20	5.334E+00	2.377E+01	1.229E+01	28

The computational results of the test instances with various scales and structures are given in Table I. It reports the minimum, maximum and mean values of RPD, and the average CPU running time (seconds) of each algorithm (T). To analyze how different performance between CMA-ES and competed algorithms, we also apply a one-way ANOVA test to confirm these mean results of RPD, where each algorithm is identified as a single factor and mean PRD as a response

variable. The overall results are shown in Table II and plotted in Fig. 6. Table II reports the source of variation including the between-groups (Columns) and the within-groups (Error), degrees of freedom (DF), the sum of squares (SS), the mean squares (MS=SS/DF), the observed value (F) based on F-statistic, and its test value (P-value). Similar to the T-tests, we apply a significance level of 0.05 for testing. Fig. 6 also provides a visual comparison based on the statistic results among algorithms.

TABLE II. ANOVA TABLE FOR RPD

Source	DF	SS	MS	F	P-value
Columns	2	1143.81	571.905	64.16	1.25173e-15
Error	60	534.82	8.914		
Total	62	1678.63			

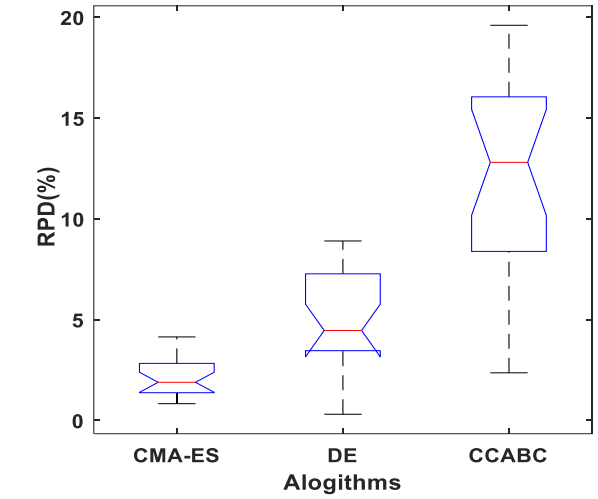

Fig. 6. Box plot of the ANOVA result of RPD

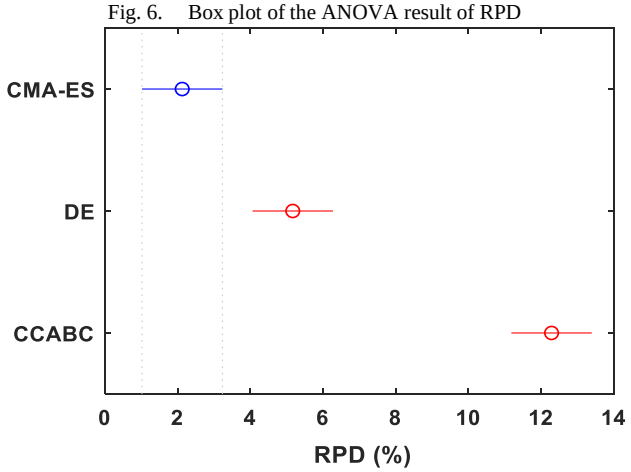

Fig. 7. Multiple comparisons on means

From the comparison results, we observe the following: (1) in comparisons of the value of PRD, the proposed CMA-ES algorithm had 17 best outputs of the given 21 test instances, which was significantly better than the other two algorithms. According to the mean value and the statistic of ANOVA, CMA-ES obtained the minimum average RPD mean value of 2.126 and significantly differed with others on RPD, since P-value=1.25e-15<0.05, and confirmed by the multiple comparisons on means (as shown in Fig. 7). Hence, CMA-ES has the best optimality ability without offline tuning. (2) Due to the complexity of matrix operations, the running time of CMA-ES is higher than other algorithms, but it is within the acceptable range (average around 1 minute).

VI. CONCLUSION

In this paper, we have studied a steelmaking scheduling problem involving time buffers which has more practical sense. We developed a MILP model to formulate the problem by considering the practical requirements on temporal, spatial, and technical factors. To solve the problem, we proposed an improved CMA-ES algorithm integrated with prior problem characteristics. It contains a novel encoding scheme, an LP-based evaluation procedure and a DE-based restart policy. Experimental results on 21 randomly synthesized instances show that the proposed CMA-ES outperforms other tested algorithms in optimality, but less competitive on efficiency.

Given these promising results, the proposed CMA-ES can be served as a Benchmarking algorithm to develop new EAs. Besides, we will explore new avenues improving the computational speed of the proposed CMA-ES algorithm to solve other types of steelmaking scheduling problems, such as with multiple objectives, under dynamic disturbances or uncertainties.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers and the editors for their constructive and pertinent comments. Sheng-Long Jiang thanks the China Scholarship Council for supporting a 1-year study at University College London.

REFERENCES

[1] L. Tang, J. Liu, A. Rong, and Z. Yang, "A mathematical programming model for scheduling steelmaking-continuous casting production," *Eur. J. Oper. Res.*, vol. 120, no. 2, pp. 423–435, Jan. 2000.

[2] L. Tang, P. B. Luh, J. Liu, and L. Fang, "Steel-making process scheduling using Lagrangian relaxation," *Int. J. Prod. Res.*, vol. 40, no. 1, pp. 55–70, Nov. 2002.

[3] D. Pacciarelli and M. Pranzo, "Production scheduling in a steelmaking-continuous casting plant," *Comput. Chem. Eng.*, vol. 28, no. 12, pp. 2823–2835, Nov. 2004.

[4] A. Atighehchian, M. Bijari, and H. Tarkesh, "A novel hybrid algorithm for scheduling steel-making continuous casting production," *Comput. Oper. Res.*, vol. 36, no. 8, pp. 2450–2461, Aug. 2009.

[5] L. Tang, Y. Zhao, and J. Liu, "An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 209–225, Apr. 2013.

[6] K. Mao, Q.-K. Pan, X. Pang, and T. Chai, "An effective Lagrangian relaxation approach for rescheduling a steelmaking-continuous casting process," *Control. Eng. Pract.*, vol. 30, pp. 67–77, Sept. 2014.

[7] J.-Q. Li, Q.-K. Pan, and K. Mao, "A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 932–949, Apr. 2015.

[8] S. Yu, T. Chai, and Y. Tang, "An effective heuristic rescheduling method for steelmaking and continuous casting production process with multirefining modes," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 46, no. 12, pp. 1675–1688, Dec. 2016.

[9] J. Long, Z. Zheng, and X. Gao, "Dynamic scheduling in steelmaking-continuous casting production for continuous caster breakdown," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3197–3216, Dec. 2017.

[10] K. Worapradya and P. Thanakijkasem, "Worst case performance scheduling facing uncertain disruption in a continuous casting process," in *Proc. 2010 IEEE Int. Conf. Ind. Eng. Eng. Manage.*, Macao, 2010, pp. 291-295.

[11] Y. Ye, J. Li, Z. Li, Q. Tang, X. Xiao, and C. A. Floudas, "Robust optimization and stochastic programming approaches for medium-term

production scheduling of a large-scale steelmaking continuous casting process under demand uncertainty," *Comput. Chem. Eng.*, vol. 66, pp. 165–185, July 2014.

[12] J. Hao, M. Liu, S. Jiang, and C. Wu, "A soft-decision based two-layered scheduling approach for uncertain steelmaking-continuous casting process," *Eur. J. Oper. Res.*, vol. 244, no. 3, pp. 966–979, Aug. 2015.

[13] S. Jiang, Z. Zheng, and M. Liu, "A multi-stage dynamic soft scheduling algorithm for the uncertain steelmaking-continuous casting scheduling problem," *Appl. Soft Comput.*, vol. 60, pp. 722–736, Nov. 2017.

[14] S. Jiang, M. Liu, and J. Hao, "A two-phase soft optimization method for the uncertain scheduling problem in the steelmaking industry," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 416–431, Mar. 2016.

[15] L. Tang, J. Guan, and G. Hu, "Steelmaking and refining coordinated scheduling problem with waiting time and transportation consideration," *Comput. Ind. Eng.*, vol. 58, no. 2, pp. 239–248, Mar. 2010.

[16] H. Missbauer, W. Hauber, and W. Stadler, "A scheduling system for the steelmaking-continuous casting process. A case study from the steel-making industry," *Int. J. Prod. Res.*, vol. 47, no. 15, pp. 4147–4172, May. 2009.

[17] Y. Tan and S. Liu, "Models and optimisation approaches for scheduling steelmaking–refining–continuous casting production under variable electricity price," *Int. J. Prod. Res.*, vol. 52, no. 4, pp. 1032–1049, May 2014.

[18] W. Xu, F. Zou, and L. Tang, "A subpopulation-based differential evolution algorithm for scheduling with batching decisions in steelmaking-continuous casting production," in *Proc. 2016 IEEE Congr. Evol. Comput.* (*CEC*), Vancouver, BC, 2016, pp. 2784–2790.

[19] A. Sbihi, A. Bellabdaoui, and J. Teghem, "Solving a mixed integer linear program with times setup for the steel-continuous casting planning and scheduling problem," *Int. J. Prod. Res.*, vol. 52, no. 24, pp. 7276–7296, May 2014.

[20] K. Mao, Q. Pan, X. Pang, and T. Chai, "A novel Lagrangian relaxation approach for a hybrid flowshop scheduling problem in the steelmaking-continuous casting process," *Eur. J. Oper. Res.*, vol. 236, no. 1, pp. 51–60, July 2014.

[21] Q.-K. Pan, L. Wang, K. Mao, J.-H. Zhao, and M. Zhang, "An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 307–322, Apr. 2012.

[22] Q.-K. Pan, "An effective co-evolutionary artificial bee colony algorithm for steelmaking-continuous casting scheduling," *Eur. J. Oper. Res.*, vol. 250, no. 3, pp. 702–714, May 2016.

[23] S. Jiang, M. Liu, J. Hao, and W. Qian, "A bi-layer optimization approach for a hybrid flow shop scheduling problem involving controllable processing times in the steelmaking industry," *Comput. Ind. Eng.*, vol. 87, pp. 518–531, Sept. 2015.

[24] J. Li, Q. Pan, K. Mao, and P. N. Suganthan, "Solving the steelmaking casting problem using an effective fruit fly optimisation algorithm," *Knowl. Based Syst.*, vol. 72, pp. 28–36, Dec. 2014.

[25] J. Del Ser et al., "Bio-inspired computation: Where we stand and what's next," *Swarm Evol. Comput.*, vol. 48, pp. 220–250, Aug. 2019.

[26] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, Mar. 2003.

[27] N. Hansen, "The CMA evolution strategy: A tutorial," arXiv preprint arXiv:1604.00772, Apr. 2016.

[28] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn Res.*, vol. 13, no. 10, pp. 281–305, 2012.

[29] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[30] H.-G. Beyer and B. Sendhoff, "Simplify your covariance matrix adaptation evolution strategy," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 746–759, Oct. 2017.

[31] C. Igel, N. Hansen, and S. Roth, "Covariance matrix adaptation for multi-objective optimization," *Evol Comput*, vol. 15, no. 1, pp. 1–28, Mar. 2007.

[32] Z. Li, Q. Zhang, X. Lin, and H.-L. Zhen, "Fast Covariance Matrix Adaptation for Large-Scale Black-Box Optimization," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 2073-2083, May 2018.

[33] Z. Li, J. Deng, W. Gao, Q. Zhang, and H.-L. Liu, "An Efficient Elitist Covariance Matrix Adaptation for Continuous Local Search in High Dimension," in *Proc. 2019 IEEE Congr. Evol. Comput.* (*CEC*), Wellington, New Zealand, 2019, pp. 936–943

[34] C.-J. Liao, E. Tjandradjaja, and T.-P. Chung, "An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem," *Appl. Soft Comput.*, vol. 12, no. 6, pp. 1755–1764, June 2012.

[35] R. Biedrzycki, "Handling bound constraints in CMA-ES: An experimental study," *Swarm Evol. Comput.*, vol. 52, p. 100627, Feb. 2020.

[36] J. Ronkkonen, S. Kukkonen, and K. V. Price, "Real-parameter optimization with differential evolution," in *2005 IEEE Congr. Evol. Comput.* (*CEC*) , Edinburgh, Scotland, 2005, pp. 506-513.