OPTIMISING STEEL PRODUCTION SCHEDULES VIA A HIERARCHICAL GENETIC ALGORITHM

K. Worapradya[1*] & P. Thanakijkasem[2]

Division of Materials Technology
School of Energy, Environment and Materials
King Mongkut's University of Technology Thonburi,
Bangkok, Thailand
[1]kiatkajohn.wora@mail.kmutt.ac.th, [2]purit.tha@kmutt.ac.th

ABSTRACT

This paper presents an effective scheduling in a steel-making continuous casting (SCC) plant. The main contribution of this paper is the formulation of a new optimisation model that more closely represents real-world situations, and a hierarchical genetic algorithm (HGA) tailored particularly for searching for an optimal SCC schedule. The optimisation model is developed by integrating two main planning phases of traditional scheduling: (1) planning cast sequence, and (2) scheduling of steel-making and timing of all jobs. A novel procedure is given for genetic algorithm (GA) chromosome coding that maps Gantt chart and hierarchical chromosomes. The performance of the proposed methodology is illustrated and compared with a two-phase traditional scheduling and a standard GA toolbox. Both qualitative and quantitative performance measures are investigated.

OPSOMMING

Effektiewe skedulering van 'n kontinue-giet staalaanleg word voorgehou. Die hoof bydrae van hierdie navorsing is die formulering van 'n nuwe optimiseringsmodel wat regte wêreld situasies beter verteenwoordig, asook 'n hiërargiese genetiese algoritme wat spesifiek aangepas word vir die soek van 'n optimale kontinue-giet skedule. Die optimiseringsmodel is onwikkel deur twee hoof beplanningsfases van tradisionele skedulering te integreer, naamlik die beplan van die giet volgorde en die skedulering van staal vervaardiging en die tydsberekening van alle werkstukke. 'n Nuwe prosedure word voorgestel vir genetiese algoritme chromosoom kodering wat Gantt-kaart en hiërargiese chromosome kombineer. Die voorgestelde metode word geïllustreer en vergelyk met 'n twee fase tradisionele skedulering en 'n standaard genetiese algoritme. Beide die kwalitatiewe en kwantitatiewe prestasiemaatstawwe is ondersoek.

* Corresponding author

1 INTRODUCTION

Production scheduling plays a critical role in improving productivity and limiting production costs, especially for mass customised production such as a steel-making-continuous casting (SCC) plant. Due to technological and economic restrictions that add extra difficulty to the SCC scheduling problem, the development of an effective and efficient methodology for scheduling an SCC plant is a challenge.

The problems related to SCC scheduling, at various production stages from steel-making to continuous casting, are characterised by two issues: how molten steel should be arranged and in what sequence, and at what time and on which machine. A review of various SCC production scheduling methods is comprehensively addressed in Tang et al. [1] and in Dutta and Fourer [2]. The SCC scheduling process was introduced by Numao and Morishita [3] and addressed in more recent literature. Numao and Morishita's [3] heuristic scheduling was developed in two steps: (1) sequencing the cast (sub-scheduling); and (2) scheduling of the steel-making process and timing of the jobs (rough scheduling), including elimination of machine conflicts (optimal scheduling). Most researchers concentrate on either step, while the rest is given. For instance, Chang et al. [4], Xue et al. [5], Jian et al. [6], and Gravel et al. [7] determined the cast sequence on continuous casters (the first step) with a binary- or integer-programming formulation, and solved it by using heuristic and artificial intelligent methods. Lee et al. [8] designed a continuous slab caster schedule via a special class of graphs called interval graphs, while Cowling et al. [9] adopted a multi-agent system for evaluating a dynamic caster schedule.

In the second step, most researchers perform the SCC scheduling by obtaining the cast sequence from a higher-level planning. Linear programming and heuristic methods are mostly adopted in this instance. Tang et al. [10] proposed an SCC scheduling that eliminates machine conflicts, via a non-linear programming model. In another work, Tang et al. [11] developed an integer-programming model and obtained the optimal solution by combining Lagrangian relaxation and a heuristic method. Pacciarelli and Pranzo [12] completed the SCC schedule by using a generalisation of the disjunctive graph of Roy and Sussman [13] and Beam search. Bellabdaoui et al. [14] focused on SCC scheduling inspired by an industrial application from the Arcelor Group to eliminate machine conflicts via linear programming and then solve them via a heuristic algorithm. In their more recent work, Bellabdaoui and Teghem [15] formulated the same problem into a mixed-integer linear programme using a commercial software package. Recently, Atighehchian et al. [16] designed SCC scheduling by using a combination of ant colony optimisation and non-linear optimisation, comparing the efficiency between a standard genetic algorithm (GA) and a heuristic method.

Several arguments emerge for taking these two steps into account together. First, the cast sequence designed in the first step often affects the optimality or creates machine conflicts. Second, the existing planning system with two-phase scheduling in many steel factories is not consistent with mass customised production. For example, a higher level planning of a compact strip plant (CSP) is not designed for short-term and variety-product planning [17]. Finally, faster scheduling and lower step planning, particularly in emergency situations, are able to stabilise the production process and reduce manpower and facility requirements.

In this paper, an efficient and effective approach is proposed that combines the two steps simultaneously into one model, so that SCC scheduling is achieved in one package. This approach focuses on the multi-production line of SCC scheduling, defined as a flexible flow shop, which is NP-complete [18]. A new mathematical model is designed by taking into account the special and practical steel requirements. Furthermore, a GA is adopted because it is particularly suitable for this practical problem. Efficiency is improved by a hierarchical chromosome coding, which can search for both optimal topology (number of jobs in each machine) and variables (starting time of jobs) of steel scheduling. Special

genetic operations are designed for discrete and continuous decision variables of the proposed model. To investigate the performance of the proposed approach, a case study is conducted on customer orders from a real factory.

2 SCC PLANT

Steel production (as shown in Figure 1) normally consists of three basic stages: melting, refining, and continuous casting. The melting stage reduces C, S, and Si in molten iron to a desirable level by burning it with oxygen in an electric arc furnace (EAF). The molten steel is then transported to a refining furnace (RF) that further refines the chemicals, eliminates impurities in the molten steel, and/or adds the required alloy ingredients. The product is delivered to the continuous caster (CC) and is poured into a tundish. The molten steel flows down through a nozzle of the tundish into a mould. It continuously solidifies into a strand (stream of steel) and is processed into steel slabs or billets.

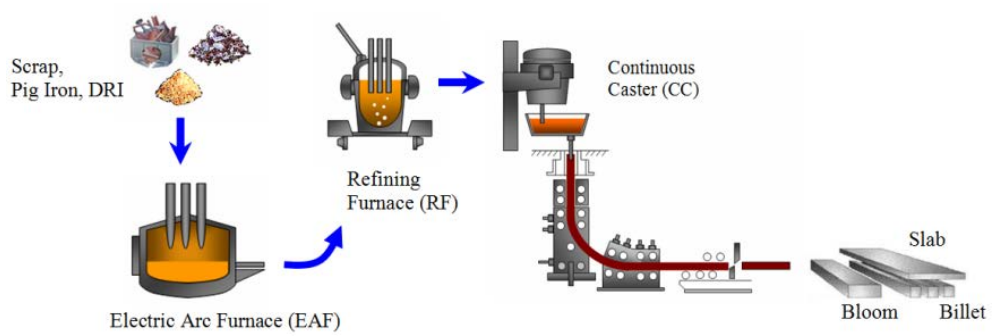


Figure 1: An illustration of SCC production

3 OPTIMISATION OF THE SCC SCHEDULE VIA A HIERARCHICAL GENETIC ALGORITHM

In this section, the mathematical optimisation model and the design of GA for SCC scheduling are proposed. The model integrates two phases of traditional planning. The GA chromosome coding and its operations are designed on a hierarchical structure to suit the proposed model.

3.1 Proposed optimisation model

The SCC scheduling problem in this paper is a multi-production line characterised as a flexible flow shop system. Two terms are introduced as follows: a *charge* (or a *job* of SCC scheduling) is a basic unit of steel-making production; and a *cast* is a set of charges casted continuously on the same CC, with a similar chemical composition. The process flow of *cast* in the parallel production line is illustrated in Figure 2. Given that molten steel is handled at high temperatures, there are strict requirements for material continuity and flow time. The problem is therefore formulated based on the following practical requirements:

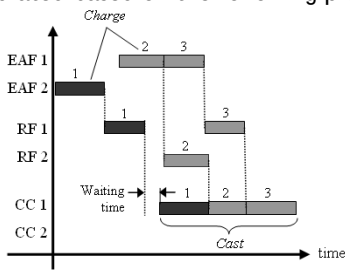


Figure 2: Process flow of a cast

Constraints

- The number of charges in a cast sequence is defined by the life-time of the nozzle at the bottom of a tundish.
- In practice, the machine setup time of each cast is required to change equipment.
- Each charge must be completely processed in operation before proceeding to the next operation.
- All charges processed on the same machine must be handled separately and sequentially.

Special requirements

The main objective chosen for the SCC scheduling model is to minimise production costs by ensuring production continuity and Just-in-Time (JIT) delivery. Several losses must be considered:

- Loss from steel grade difference in adjacent charges: a slab mixed with another steel grade will be assigned to the second grade product.
- Loss from width and thickness change: the material between the changes of charges will be disposed of.
- Loss from consignment date (e.g., inventory and compensation to customers and shipping).
- Loss from cast break: the cast break causes the remaining molten steel to be re-heated or drained out.
- Loss from waiting time: a drop in the molten steel's temperature because of the waiting time affects the quality of the steel.

The optimisation model is formulated by the above conditions and inspired by Tang et al. [10]. A mixed–integer programming model is used to represent the production costs as follows:

$$
\begin{aligned}
f(\mathbf{X},\mathbf{Y},st) \;=\; & \frac{1}{2}\left(\sum_{m=1}^{M_C}\sum_{k\in P_m}\mathbf{X}^{(k)\mathrm{T}}\mathbf{Q}_C\mathbf{X}^{(k)}+\sum_{m=1}^{M_F}\mathbf{Y}^{(m)\mathrm{T}}\mathbf{Q}_F\mathbf{Y}^{(m)}\right)\\
& +\sum_{k=1}^{P}\sum_{i,\,j\in\Omega_k}C_B*(st_j^{(m)}-st_i^{(m)}-T_i^{(m)})\\
& +\sum_{i=1}^{N}\sum_{m,n\in\Pi_i}C_W*(st_i^{(m)}-st_i^{(n)}-T_i^{(n)})\\
& +\sum_{i=1}^{N}\sum_{m\in\Pi_C}C_L*\max(0,\,st_i^{(m)}+T_i^{(m)}-d_i)\\
& -\sum_{i=1}^{N}\sum_{m\in\Pi_C}C_E*\min(0,\,st_i^{(m)}+T_i^{(m)}-d_i)
\end{aligned}
\tag{1}
$$

subject to:

- for binary variables (Assignment)

$$\sum_{k=1}^{P}x_i^{(k)}=1 \tag{2}$$

$$\sum_{m=1}^{M_E}y_i^{(m)}=1 \quad \text{and} \quad \sum_{m=(M_E+1)}^{M_F}y_i^{(m)}=1 \tag{3}$$

$$2\le\sum_{i=1}^{N}x_i^{(k)}\le L \tag{4}$$

$$x_i^{(k)},\,y_i^{(m)}\in\{0,1\} \tag{5}$$

- for continuous variables (Time relations)

Melting and Refining ($m \in \Pi_F$)

$$st_i^{(m)} \geq st_i^{(n)} + T_i^{(n)} \quad , \ m,n \in \Pi_i \tag{6}$$

$$st_j^{(m)} \geq st_i^{(m)} + T_i^{(m)} \quad , \ m \in \Pi_i \tag{7}$$

Continuous Casting ($m \in \Pi_C$)

$$st_j^{(m)} \geq st_i^{(m)} + T_i^{(m)} + \delta^{(m)} \quad , \ i \in \Omega_k, j \in \Omega_{k+1} \tag{8}$$

$$st_j^{(m)} = st_i^{(m)} + T_i^{(m)} \quad , \ i,j \in \Omega_k \tag{9}$$

where

$$\mathbf{X}^{(k)} = \begin{bmatrix} x_1^{(k)} \\ \vdots \\ x_N^{(k)} \end{bmatrix}, \quad \mathbf{Y}^{(k)} = \begin{bmatrix} y_1^{(m)} \\ \vdots \\ y_N^{(m)} \end{bmatrix} \tag{10}$$

$$\mathbf{Q}_C = \begin{bmatrix} q_{1,1} & \cdots & q_{1,N} \\ \vdots & \ddots & \vdots \\ q_{N,1} & \cdots & q_{N,N} \end{bmatrix}, \ \mathbf{Q}_F = \begin{bmatrix} c_{1,1}^G & \cdots & c_{1,N}^G \\ \vdots & \ddots & \vdots \\ c_{N,1}^G & \cdots & c_{N,N}^G \end{bmatrix}, \ q_{i,j} = c_{i,j}^G + c_{i,j}^{WD} + c_{i,j}^{TH} \tag{11}$$

$$c_{i,j}^G = \begin{cases} 0 & \text{charge } i \text{ and } j \text{ are the same steel grade serial} \\ f_1 & \text{charge } i \text{ and } j \text{ belong to the same steel grade serial} \\ f_2 & \text{charge } i \text{ and } j \text{ are the different steel grade serial} \end{cases} \tag{12}$$

$$c_{i,j}^{WD} = f_3 * (\Delta wd_{i,j})^2, \quad \Delta wd_{i,j} = wd_i - wd_j \tag{13}$$

$$c_{i,j}^{TH} = f_4 * (\Delta h_{i,j})^2, \quad \Delta h_{i,j} = h_i - h_j \tag{14}$$

$$i, j \in \Omega \text{ and } j \text{ follows } i \tag{15}$$

$$m,n \in \Pi \text{ and } m \text{ follows } n \tag{16}$$

$$k = 1, \cdots, P \tag{17}$$

Decision variables

- $\mathbf{X}^{(k)}$ is a binary decision vector to allocate the charge into cast k on CC. Charge i belongs to cast k if and only if $x_i^{(k)}$ is equal to 1.
- $\mathbf{Y}^{(m)}$ is a binary decision vector to allocate the charge into EAF and RF m ($m \in \Pi_F$). Charge i is operated on machine m if and only if $y_i^{(m)}$ is equal to 1.
- $st_i^{(m)}$ is the starting time of the charge i on machine m .

Other notations

- $\mathbf{Q}_C$ is a cost matrix for CC and $\mathbf{Q}_F$ is a cost matrix for EAF and RF. $q_{i,j}$ is the production cost between charge i and j due to losses as follows: $c_{i,j}^G$, $c_{i,j}^{WD}$, and $c_{i,j}^{TH}$ are losses of steel grade difference, loss of width change, and loss of thickness change, respectively.
- Π is a set of all machines, $\Pi = \{1, 2, \ldots, M\}$, where M is the total number of machines.
- Π_F is a set of steel-making furnaces (EAF and RF), $\Pi_F = \{1, 2, \ldots, M_F\}$, where M_F is the total number of EAF and RF. $\Pi_F \subset \Pi$. M_E is the total number of EAF and M_R is the total number of RF, where $M_E + M_R = M_F$. Π_C is the set of CC, $\Pi_C = \{1, 2, \ldots, M_C\}$, where M_C is the total number of casters. $M_C + M_F = M$, $\Pi_C \subset \Pi$, and $\Pi_C \cap \Pi_F = \varnothing$. Π_i is the set of machines used for charge i , $\Pi_i \subset \Pi$.

- P is the total number of casts and P_m is the set of casts that are processed on caster m. For example, $P_1 = \{1, 3\}$ means that cast number 1 and 3 are operated on caster number 1.
- Ω is the set of all charges, $\Omega = \{1,2,..., N\}$, where N is the total number of charges to be arranged. Ω_k is the set of charges in cast k.
- L is the maximum number of charges in a cast.
- $f_1,\cdots, f_4$ are penalty factors.
- wd_i is the ordered slab width of charge i. h_i is the ordered slab thickness of charge i and d_i is the consignment date or time of charge i.
- $T_i^{(m)}$ is the processing time of charge i on machine m. $\delta^{(m)}$ is the setup time of caster m.
- C_W, C_B, C_L and C_E are the penalty factors of waiting time (*W*), cast break time (*B*), lateness (*L*), and earliness (*E*), respectively.

The last four terms in Equation (1) represent the cast break loss, waiting time, lateness, and earliness penalties, respectively. While the real decision variable ($st_i^{(m)}$) is used for timing the charges on the machines, it also implies the charge sequence. The constraint in Equation (2) ensures that every charge must be arranged to a cast, while the constraint in Equation (3) ensures that every charge must be arranged once to a machine in each process. Equation (4) ensures that the number of charges in each cast must not be less than two and cannot exceed the endurance capability of each tundish. Equation (6) ensures that the two consecutive operations of each charge are executed sequentially. Equation (7) ensures that two contiguous charges are not processed simultaneously on the same machine. Equation (8) ensures that enough setup time is required between casts, while the constraint in Equation (9) ensures the continuity of casting.

3.2 Designing a Hierarchical Genetic Algorithm

Since gradient-based optimisation methods are not practical for finding an optimal schedule for the binary- or integer-programming model, this work applies a GA that is widely used, especially in flow shop production [19]. In a real-life process, a GA can be easily modified with respect to the objectives and constraints [17, 20]. Because this problem focuses on a particular type of application, this work adopts the concept of a Hierarchical Genetic Algorithm (HGA), because it provides suitable GA operations to avoid infeasible solutions.

3.2.1 Hierarchical chromosome coding

A hierarchical chromosome coding for GAs is introduced in Tang et al. [21]. This concept is regarded much as DNA is regarded in a biological chromosome. DNA consists of two types of genes: (1) structural genes and (2) regulatory genes. The structural genes contain the genetic information, while the regulatory genes control coding of the structural genes. Similarly, a mathematical hierarchical chromosome can be classified into two different types: i(1) parametric genes, and (2) control genes. The parametric genes, which contain parametric values of the objective function, are analogous to the structural genes. The activation of the parametric genes is governed by the control genes, which are analogous to the regulatory genes. In Figure 3, these three-level gene structures are illustrated. The value 0 or 1 of the control genes is used to define the activation of the parametric genes.

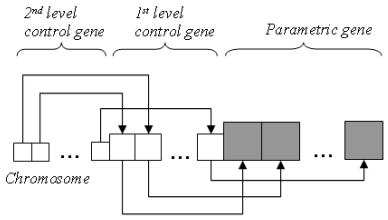


Figure 3: General form of the hierarchical chromosome

From the proposed optimisation model, the decision variables, $\mathbf{X}^{(k)}$, $\mathbf{Y}^{(m)}$, and $st_i^{(m)}$ are coded into two types of chromosomes as follows:

- ***Variable chromosome*** **(**H_v**)**: The variable chromosome comprises a job-control gene and a starting-time gene. The job-control gene (g_c) contains the binary variable $\mathbf{X}^{(k)}$ or $\mathbf{Y}^{(m)}$ while the starting-time gene (g_s) contains the real variable $\Delta t_i^{(m)}$, which is a slack of the starting-time according to Equation (7) or $st_j^{(m)} = st_i^{(m)} + T_i^{(m)} + \Delta t_j^{(m)}$, j follows i. The value 1 of the job-control gene will activate the associated starting-time gene and address the number of charges on machine m as shown in Figure 4.

$$g_s = \{\Delta t_1^{(1)}, \Delta t_2^{(1)}, \cdots, \Delta t_N^{(1)}, \quad \Delta t_1^{(2)}, \cdots, \Delta t_N^{(2)}, \quad \Delta t_1^{(m)}, \cdots \Delta t_N^{(m)}\} \tag{18}$$

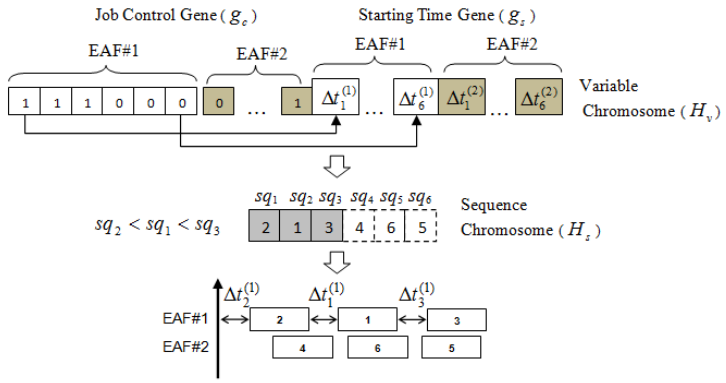


Figure 4: The proposed hierarchical chromosome

- ***Sequence chromosome*** **(**H_s**):** the sequence chromosome arranges the activated starting-time gene. It represents a set of integer numbers as follows:

$$H_s = \{h^{(E)}, h^{(R)}, h^{(C)}\} \tag{19}$$

$$h^{(E \text{ or } R \text{ or } C)} = \{sq_i \mid sq_i \in [1, N] \text{ and } sq_i \neq sq_j, \quad i, j \in \Omega\} \tag{20}$$

where $h^{(E)}$, $h^{(R)}$, and $h^{(C)}$ are the sub-groups of the sequence chromosome of EAF, RF, and CC, respectively. Each integer number (sq_i) shows the priority of each member ($\Delta t_i^{(m)}$) of the starting-time gene. An activated charge with the smallest integer number will be processed first (shown in Figure 4). Figure 4 illustrates the chromosomes coding and decoding for the first melting machine (EAF#1). The job control gene is the variable $\mathbf{Y}^{(1)}$, in which the first three charges are enabled. Therefore the slack $\Delta t_1^{(1)}$, $\Delta t_2^{(2)}$, and $\Delta t_3^{(3)}$ are selected. These activated slacks are then sequenced by the sequence chromosome, e.g., $sq_2 < sq_1 < sq_3$. Consequently, charge 2 is executed and is then followed by charge 1 and 3, sequentially. It is noted

that the sequence of the activated charges for EAF#1 will be different if the priority in the sequence chromosome is different.

3.2.2 Genetic operations

The main operation of the HGA optimisation consists of selection, crossover, and mutation. Since there are two types of genes in the chromosome, a specific crossover and mutation method has been designed to suit this particular purpose.

- ***Variable chromosome crossover***: The crossover is performed separately for two types of genes. To avoid infeasible solutions, according to Equation (2), first a parent chromosome of the control genes is formed by arranging all job-control genes into a matrix. A one-point crossover with a probability rate is then performed by swapping the matrix columns, as shown in Figure 5. The offspring A is made by replacing the last two columns behind the crossover point of the parent A with the last two columns of the parent B. Since the starting-time gene is a vector of real numbers, the standard one or multiple-point crossovers can be directly applied.
- ***Sequence chromosome crossover***: The standard one-point crossover is adopted by randomly choosing the fixed crossover points a and b, as shown in Figure 6. An example shows that when crossover point a is chosen, offspring A is consequently created from swapping $h^{(R)}$ and $h^{(C)}$ between parent A and parent B.
- ***Mutation***: For variable chromosomes, a bit mutation is applied to the job-control gene, while the random mutation shown in Equation (21) is applied to the starting-time gene.

$$\Delta t_i^{(m)} = \Delta t_i^{(m)} + \psi(\mu, \sigma) \tag{21}$$

where ψ is a random function (may be normally distributed), μ and σ^2 are mean and variance, respectively. For the sequence chromosome, a special mutation operator has been designed to find an optimal sequence. A delta-shift mutation [22], which alters each element in the sequence chromosome, is adopted as follows:

$$sq_i^{(m)} = sq_{i+\Delta i}^{(m)} \tag{22}$$

where Δi has equal chance to be 1 or -1 with a small probability.

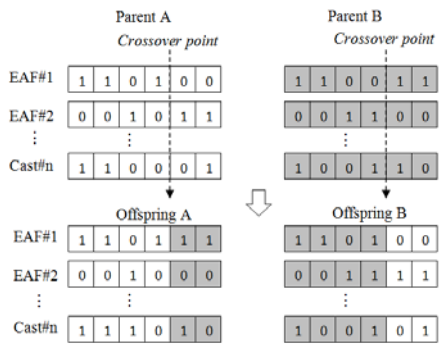


Figure 5: Crossover for variable chromosome

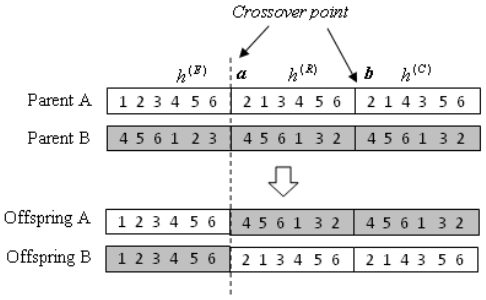


Figure 6: One-point crossover for sequence chromosome

4 COMPUTATIONAL EXPERIMENT

To verify the proposed optimisation model and test the performance of the methodology, this section presents a case study of scheduling in an SCC factory that consists of two EAFs, two RFs, and two CCs. A daily production lot contains 12 charges (as shown in Table 1), with the factory wanting to arrange these orders optimally into four casts (grade groups). The first and second casts are processed on caster 1, while the third and forth casts are

processed on caster 2. The related set in the model can be defined as follows: $\Pi = \{1,2,3,4,5,6\}$, $\Pi_F = \{1,2,3,4\}$, $\Pi_C = \{1,2\}$, $\Omega = \{1,2,...,12\}$.

The experiments were implemented on MATLAB and carried out on a 1.66 GHz Intel(R) Core 2 CPU personal computer with 4 GB memory. The processing time of SCC production was defined by a real factory situation. The system performance was defined in terms of the production costs in US dollars (USD). The EAF and RF processing times were defined exactly at 50 minutes, while a dynamic processing time was used for CC. The casting time ($T_i^{(m)}$) was calculated from slab dimension, casting speed (v_i), and steel weight (W_{ladle}) in the ladle, as expressed in Equation (23).

$$T_i^{(m)} = \frac{W_{ladle}}{v_i w_i h_i \rho_s} \quad , \quad m \in \Pi_C \tag{23}$$

where ρ_s is steel-specific weight, w_i and h_i are slab width and thickness of charge i. The cast speed is between 4.4-5.5 m/min, and depends on steel grade and mould dimension.

Table 1: Production orders

Charge No.	Due time (min)	Grade serial	Factory steel grade	Width (mm)	Thickness (mm)	Weight (ton)
1	320	SM400A	1008MnDK2	1272	60	14300
2	320	SM400A	1008MnDK2	1258	60	14300
3	320	SM400A	1008MnDK2	1252	50	14300
4	320	SS400	1008MnDK-M1	1230	60	16000
5	320	SS400	1008MnDK-M1	1230	60	16000
6	350	SS400	1008MnDK-M3	1272	60	24500
7	420	SS400	1008MnDK-M3	1552	50	14300
8	450	SS400	1008MnDK-M3	1245	50	16000
9	450	SM400A	1008MnDK2	1240	60	16000
10	480	SPHC	1007DKGXA-00	1272	60	14300
11	480	SPHC	1007DKGXA-00	1255	60	19500
12	500	SPHC	1007DKGXA-00	1275	50	24500

4.1 Model performance

Focusing on the performance of the proposed optimisation model, the scheduling is optimised by using the proposed model with a standard GA (a MATLAB tool). The result is compared with the 2-phase traditional scheduling that an initial cast sequence is defined (based on Xue et al. [5]) and then timing of all jobs is assigned. The model parameters, as presented in Table 2, are based on a real factory situation. The approximate penalty factors are defined from the production costs in USD.

The best results (from ten trials) using the proposed model and the two-phase traditional approach are presented in a Gantt chart in Figures 7 and 8 respectively. The performance comparison is shown in Table 3, which reveals that the proposed model can provide a better total cost than the traditional approach. It is observed that, although the traditional method can provide a somewhat better cast sequence plan in the first phase (considering only the cost from the grade mixing, size changing, and consignment sequencing at CC), the total cost becomes undesirable in practice when the timing of all processes is assigned in the second phase.

Table 2: Optimisation model and GA parameters

Optimisation model		GA	
Parameters	Value	Parameters	Value
f_1, f_2, f_3, f_4	600, 14400, 0.07, 1.6	Representation	Binary
C_W, C_B, C_L, C_E	6.7, 14400, 16.2, 5.4	Pop. Size/Max. iter.	600/2000
N	12	Crossover	2 point (rate 0.8)
P	2	Selection	Stochastic uniform
L	4	Mutation	Bit (rate 0.01)

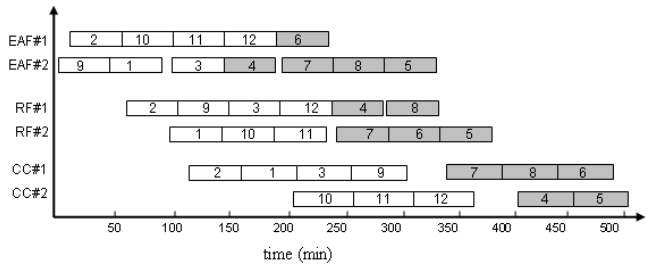


Figure 7: Best schedule from the proposed model

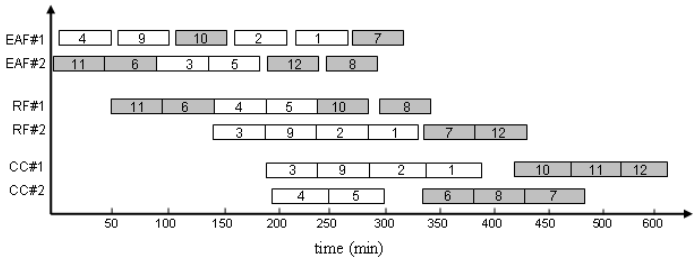


Figure 8: Best schedule from the two-phase traditional approach

Table 3: Comparison of results (the proposed model vs the two-phase traditional approach)

Objective factors	Performance (per production line)	
	Proposed model	2-Phase traditional approach
Optimal objective value ($\times 10^4$ USD)	2.16	2.41
Steel grade mixing (ton)	0	0
Width/ thickness change (ton)	12.8	10.4
Total consignment date delay (min)	218	226
Total cast break (min)	0	0
Total waiting time (min)	118	223
Total completion time (min)	506	619

4.2 Optimisation performance

The performance of the proposed methodology, which combines the proposed model and an HGA, is compared with the standard GA in this section. It is known that the quality of the optimal solution from the GA frequently depends on the population size, and the same initial solution does not always provide the same result over different searches due to uncertainty in the method. In practice, the factory needs an approach that is reliable and that provides an acceptable result within an appropriate computational time. Thus the experiment is divided into two cases: population sizes of 400 and 600 respectively. In each population size, five different test sets (initial set no.1-5) will be executed repeatedly –

five times in each test set (a total of 25 samples) to observe the variation. The HGA and GA parameters are shown in Table 4.

Table 5 shows the optimisation results from observing the overall performance by comparing the average results from each population size through 25 trials. Regarding the qualitative assessment, the proposed approach can provide a lower average than the standard GA does for all population sizes. The percentage differences are around 6.6% and 5.7% for the 400 and 600 population sizes respectively. Moreover, the proposed approach provides a smaller standard deviation. The results of all 25 trials are shown in Figure 9. It can be claimed that the proposed approach is more likely to provide a better solution than the standard GA does.

Table 4: HGA and GA parameters for methodology demonstration

GA Parameters	Proposed methodology		Standard GA
	H_v	H_s	
Representation	Binary, Real	Integer	Binary
Population size	400, 600	400, 600	400, 600
Max. iteration	2000	2000	2000
Crossover	1 point (rate 0.8)	1 point (rate 0.8)	2 point (rate 0.8)
Selection	Stochastic uniform	Stochastic uniform	Stochastic uniform
Mutation	Random (rate 0.01)	Delta shift (rate 0.01)	Bit (rate 0.01)

Table 5: Optimisation results

Pop. size	Algorithm	Average		
		Objective value ($\times 10^4$ USD/Lot)	σ	%Diff*
400	Standard GA	3.14	0.56	-5.7
	Proposed GA	2.96	0.45	
600	Standard GA	3.05	0.58	-6.6
	Proposed GA	2.85	0.35	

* %Diff = $100 \times (Obj_{pro} - Obj_{std}) / Obj_{std}$

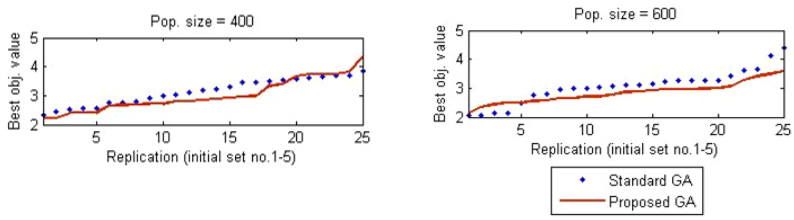
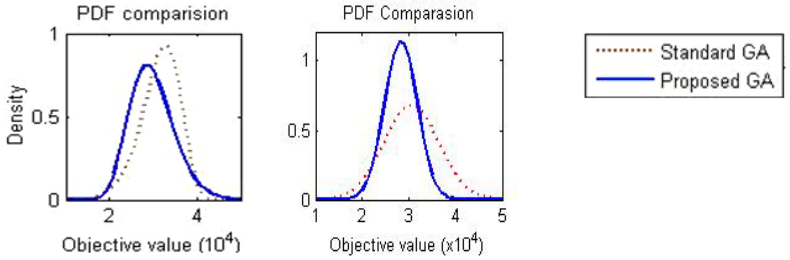


Figure 9: Quantitative comparison of all results shown by population size

Furthermore, the quantitative performance of the proposed approach is evaluated via a probabilistic assessment. The probability density functions (PDF) of both approaches are obtained from the results of 25 trials in each population size, as shown in Figure 10 (a) and (b). The achievable probability is calculated and compared in Table 6. It implies that the proposed approach is likely to achieve an acceptable solution about 70 per cent of the time, while the standard GA scores only 55 per cent and 48 per cent for the population sizes of 400 and 600 respectively. In other words, the proposed approach is more likely to provide a better result.



(a) Pop. size = 400 (b) Pop. size = 600

Figure 10: Plot of average of best objective value in each initial set

Table 6: Comparison of probability of the method achievement

Pop. size	Proposed GA	Standard GA	$\%Eff = 100 \times (P_{pro}(X) - P_{std}(X)) / P_{std}(X)$
400	$P(X \le 32510) = 0.70$	$P(X \le 32510) = 0.55$	27.2
600	$P(X \le 30200) = 0.70$	$P(X \le 30200) = 0.48$	45.8

Regarding the computational time, the average time of all cases is shown in Table 7. It can be seen that the standard GA needs around five times more computational time than the proposed approach. It is presumed that the better computational time for the proposed approach arises because the design better fits the particular scheduling purpose, while the standard GA aims for general purpose scheduling.

Table 7: Average of computational time consumption

Pop. size	Average of computational time (min)	
	Standard GA	Proposed GA
400	193.3	30.4
600	280.0	42.0

5 CONCLUDING REMARKS

This paper proposes a novel scheduling methodology for an SCC production (a flexible flow shop production) that is NP-complete. The methodology can optimise both caster sequencing and steel-making scheduling phases simultaneously via a proposed optimisation model. The HGA with specific operations is introduced to search for the optimal schedule. The efficiency of both the proposed optimisation model and the methodology is illustrated by the simulation of the real-world case. The experimental results are compared with the standard GA against three criteria; (1) the quality of the solution, (2) the quantity of achievement in terms of the probabilistic assessment, and (3) the computational time needed. It can be seen that the proposed methodology can satisfy all criteria, especially in a larger population.

It is noted that this case study, along with many steel factories, is based on a compact strip plant (CSP) design, which has no buffer between the steel-making plant and the rolling mill. A JIT concept is therefore necessary. Consequently, the earliness and lateness penalties in the optimisation model are practically emphasised, and the consignment date is set as the due time for delivering the slabs to the rolling mill. It should be noted that the application of this methodology to other steel production types producing only slabs or billets can be slightly different. In slab or billet factories, the earliness penalty is often relaxed or approximated from the warehouse cost, while the lateness penalty is approximated from the marine cargo delay expense. In addition, the cast break penalty may be relaxed in some cases. In practice, the caster operators can avoid cast breaks by reducing the casting speed while awaiting the charge from the RF. However, this method (reducing casting speed) generally works only when the waiting time is less than 30 minutes due to the steel degradation.

ACKNOWLEDGEMENT

This work was partially supported by the Higher Education Research Promotion and National Research University Project of Thailand, Office of the Higher Education Commission.

REFERENCES

[1] **Tang, L.X., Liu, J., Rong, A. & Yang, Z.** 2001. A review of planning and scheduling systems and methods for integrated steel production. *European Journal of Operational Research*, 133, pp 1-20.

[2] **Dutta G. & Fourer R.** 2001. A survey of mathematical programming application in integrated steel plants. *Manufacturing & Service Operations Management*, 3 (4), pp 387-400.

[3] **Nuamo, M. & Morishita, S.I.** 1991. Cooperative scheduling and its application to steelmaking processes. *IEEE Transactions on Industrial Electronics*, 38 (2), pp 150–155.

[4] **Chang, S.Y., Chang, M.-R. & Hong, Y**. 2000. A lot grouping algorithm for a continuous slab caster in an integrated steel mill. *Production Planning & Control*, 11, pp 363-368.

[5] **Xue, Y., Yang Q. & Shao, H.** 2004. Optimum cast plan for steelmaking-continuous casting production scheduling. *IEEE International Conference on Control Applications*, pp 1394-1397.

[6] **Jian, W., Xue, Y.C. & Yang, Q.W.** 2004. Optimum cast plan for steelmaking-continuous casting production scheduling using artificial fish swarm optimization algorithm. *3rd International Conference on Machine Learning and Cybernetics*, pp 2339-2341.

[7] **Gravel, M., Price, W. & Gagné, C.** 2002. Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of Operational Research*, 143, pp 218-229.

[8] **Lee, K., Chang, S.Y. & Hong, Y.** 2004. Continuous slab caster scheduling and interval graphs. *Production Planning & Control*, 15, pp 495-501.

[9] **Cowling, P.I., Ouelhadj, D. & Petrovic, S.** 2004. Dynamic scheduling of steel casting and mill using multi-agents. *Production Planning & Control*, 15 (2), pp 495-501.

[10] **Tang, L.X., Luh, P.B., Liu, J. & Fang, L.** 2000. A mathematical programming model for scheduling steel-making-continuous casting production. *European Journal of Operational Research*, 120, pp 55-70.

[11] **Tang, L.X., Luh, P.B., Liu, J. & Fang, L.** 2002. Steelmaking process scheduling using Lagrangian relaxation. *International Journal of Production Research*, 40, pp 55-70.

[12] **Pacciarelli, D. & Pranzo, M.** 2004. Production scheduling in a steelmaking-continuous casting plant. *Computers and Chemical Engineering*, 28, pp 2823–2835.

[13] **Roy, B., & Sussman, B.** 1964. Les problemes d'ordonnancement avec contraintes disjonctives. *Note DS No. 9bis. Paris: SEMA*.

[14] **Bellabdaoui, A., Fiordaliso, A. & Teghem, J.** 2005. A heuristic algorithm for scheduling the steel making continuous casting process. *Pacific Journal of Optimization*, 1, pp 1-18.

[15] **Bellabdaoui, A. & Teghem, J.** 2006. A mixed-integer linear programming model for the continuous casting planning. *International Journal Production Economics*, 104, pp 260-270.

[16] **Atighehchian, A., Bijari, M. & Tarkesh, H.** 2009. A novel hybrid algorithm for scheduling steel-making continuous casting production. *Computer and Operations Research*, 36, pp 2450-2461.

[17] **Bierwirth, C. & Mattfeld, D.C.** 1999. Production scheduling and rescheduling with genetic algorithms. *Evolutionary Computation*, 7 (1), pp 1-17.

[18] **Gupta, J.N.D.** 1988. Two state hybrid flow shop scheduling problems. *Operational Research*, 39, pp 359-364.

[19] **Reeves, C.A.** 1995. A genetic algorithm for flow shop scheduling. *Computers and Operation Research*, 22, pp 5-13.

[20] **Bürvenich, H.P**. 1999. CSP sequence planning and optimization. *SIEMENS Metals, Mining, & More*, pp 4-5.

[21] **Tang, K.S., Man, K.F., Liu, Z.F. & Kwang, S.** 1998. Minimal fuzzy memberships and rules using hierarchical genetic algorithms. *IEEE Transaction on Industrial Electronics*, 45 (1), pp 162-169.

[22] **Man, K.F., Tang, K.S. & Kwong, S.** (1999). *Genetic algorithms: Concepts and designs*. London: Springer-Verlag.