Available online at www.sciencedirect.com
SCIENCE DIRECT®
Int. J. Production Economics 100 (2006) 30–43
international journal of production economics
www.elsevier.com/locate/ijpe

Order-planning model and algorithm for manufacturing steel sheets

Shixin Liu[a,*], Jiafu Tang[a], Jianhai Song[b]

[a]*School of Information Sciences & Engineering, Northeastern University, Shenyang 110004, PR China*
[b]*Metallurgical ERP Business Department, Shanghai Baosight Software Company, Ltd., Shanghai 201900, PR China*

Received 12 June 2003; accepted 1 October 2004
Available online 26 November 2004

Abstract

A multi-objective order-planning model is formulated for manufacturing steel sheets. The objectives include minimizing tardiness cost, balancing utility of capacities and minimizing inventory cost. The weighted-sum approach is used for transferring the multiple objectives into single one, as well as a penalty function is used for incorporating constraints into the objective function. Specific particle swarm optimization (PSO) algorithm is designed to solve the model. Taking three practical order-planning problems as instances, different sets of parameter combinations are systemically designed to test effectiveness and efficiency of the algorithm in experiments; computation results show that the model and algorithm are superior to human–machine coordination method.

Keywords: Iron–steel industry; Steel sheet manufacturing; Flow shop process; Order planning; Weighted sum of criteria; Particle swarm optimization (PSO)

1. Introduction

Within the iron–steel industry, global competition and rapidly changing customer requirements are becoming more drastic; several challenges are evident which include responding effectively to customer requests, maximizing plant throughputs, reducing work in progress/finished stock levels and reducing manufacturing/distribution costs. The search continues for innovative ways to improve manufacturing responsiveness through increasingly effective management of order fulfillment processes. Making more reasonable and efficient production plans is a useful way, at least in part, to meet the pressure faced by enterprises. The production is usually planned based on customer orders in most of iron–steel plants; its production planning includes order planning and daily production planning. Order planning aims to assign

*Corresponding author. Tel.: +86 24 836 80 673.
E-mail address: sxliu@mail.sy.ln.cn (S. Liu).

0925-5273/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.ijpe.2004.10.002

finish time of each process of orders according to due date, production capacities, and other constraints, so as to keep the production line running smoothly. The daily production planning, namely shop floor planning, includes making heat plan, cast lot sizing plan and rolling batch plan according to released order plans and production process programs (Tang et al., 2000a, b; Tamura et al., 1998). Comparing with daily production planning, little attentions are paid to order planning by researchers and practitioners. However, because of its particularities and key roles in iron–steel production process, order plan has an important effect on the compiling of daily plan; the more suitable and efficient the order plan is, the easier the compiling of daily plan. Contrarily, if an order plan is not suitable, it would be very difficult to make daily plan, at the worst, no feasible daily plan could be found.

Redwine and Wismer (1974) formulated a mix integer-programming model with the objective of minimizing total tardiness cost for order-planning problem in the steel–iron plant. Zhang et al. (2000) presented an integer-programming model with the objective of minimizing total earliness and tardiness cost for the order planning, and designed a genetic algorithm based on repeatable natural scale code and 3-mutation operator to solve the model. Computational results showed that the solutions obtained by the model were superior to those obtained by the human–machine system. Sasidhar and Achary (1991) modeled the production-planning problem in a steel mill with the objective of maximizing capacity utilization as a maximal flow problem in a Multiple Arc Network (MAN). The model took into account the priorities assigned to the customers and also the order balance positions. An algorithm was presented for solving the MAN formulation with customer priorities. Portmann and Rohr (1995) designed a hierarchical approach consisting of two levels for the medium planning and the short-term scheduling problem in iron–steel plants. In order to solve the scheduling problems arising from this approach, several methods were proposed. Cohen et al. (1997) introduced the Oregon Steel Mills (OSM) Order Application System (OASYS) blended expert systems methodology and numerical optimization technology to maximize the hot mill slab inventory value. The system's underlying technologies, its current status, and how it would operate were examined. Fujii et al. (1996) developed a consistent scheduling system for sheet metal production, which had (1) a long-term rough scheduling function, (2) a short-term detail scheduling function, and (3) a schedule modification function. These three functions made it possible to estimate the work load, the amount of inventories and the delivery date of each customer order within a month. Moreover, it had become possible to reduce schedule planning time and to show accurate delivery dates.

In summary, most of the models for order planning in iron–steel manufacturing are single-objective focused. While in practical production management, production manager faces more than one objective, which should be balanced, multi-objective formulation may be more accepted by a practitioner. In this paper, based on studies of the order-planning strategies, constraints and objectives in iron–steel plants under new global competition environment, a multi-objective order-planning model (OPM) for steel sheets manufacturing is presented. The weighted-sum approach is used for transferring the multi-objective model into a single-objective one. Specific particle swarm optimization (PSO) algorithm is designed to solve it.

The rest of this paper is organized as follows. Section 2 describes the general manufacturing process and the order-planning problems in a large-scale iron–steel plant. A new OPM and a modified PSO algorithm for solving the model are described in Sections 3 and 4, respectively. Section 5 reports computational results in experiments. Finally, conclusions and future research directions are drawn in Section 6.

2. Problem description

2.1. General manufacturing process

As illustrated in Fig. 1, steel sheets manufacturing processes from raw materials could mainly be decomposed into six processes: (1) a blast furnace (BF), (2) a basic oxygen furnace (BOF), (3) continuous

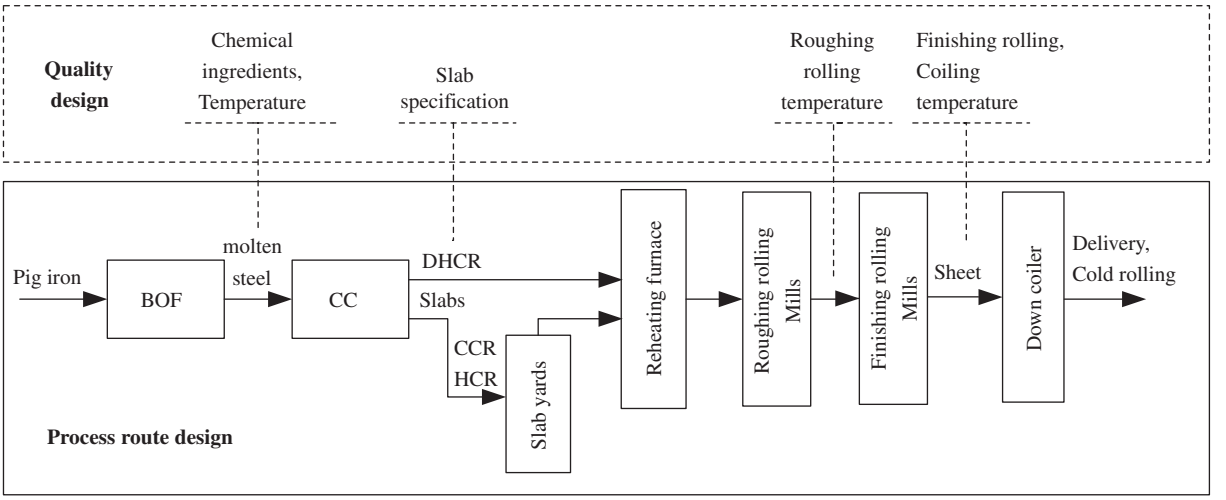

Fig. 1. Steel sheets manufacturing process.

casters (CCs), (4) a hot rolling mill (HRM), (5) a cold rolling mill (CRM), and (6) a surface treatment process. The process BF makes pig iron, which is poured into the BOF, where it transforms into molten steel. The process CC transforms molten steel into slabs with different sizes. The HRM and the CRM are used to roll and coil them, then the surface treatment is performed over steel sheets.

The whole manufacturing process is described as follows. First, iron ore, coke and limestone are fed into the BF, then the iron ore is reduced and smelted. Such iron is called pig iron. Second, the pig iron is transformed into molten steel in the BOF by blowing pure oxygen and removing impurities. After ladle refining, molten steel is solidified into slabs through the process CC. According to different charge technologies, such as Cold Charge Rolling (CCR), Hot Charge Rolling (HCR), and Direct Hot Charge Rolling (DHCR), slabs are reheated in the reheating furnace (RF), and then transferred to the HRM, where they are rolled hot. A rolled slab, called a sheet, is coiled hot by a down coiler. Some hot coils are sold to customers as they requested. The remaining coils are transferred to the cold rolling mill to turn out cold-rolled sheets, and after this process, given an immaculate surface.

2.2. Production organization process

During the steel sheets manufacturing processes, production organization process and order planning are important and core activities which are discussed in this section. In iron–steel plants, production process is typically organized as illustrated in Fig. 2. After receiving *user orders*, the sales management system checks the feasibility of the orders according to rough production capacities plans, financial criteria, and process program constraints, then releases the feasible orders to production and quality management system, where the process routes and quality criteria are designed for the orders. The process route design is a process of optimizing the manufacturing routes for orders so as to suggest a manufacturing route with short production cycle, high gain rate and feasible production capacities, taking into account order requirements, enterprise manufacturing standards, current production capacities and conditions. Quality design aims to establish quality criteria and specifications of works in process (WIP) and final products in every process through production line based on selected manufacturing routes so that the order requirements for final products could be met. These quality criteria and specifications include chemical ingredients of molten steel, quality and size of slabs, temperature of roughing rolling, temperature of finishing rolling, etc. The results

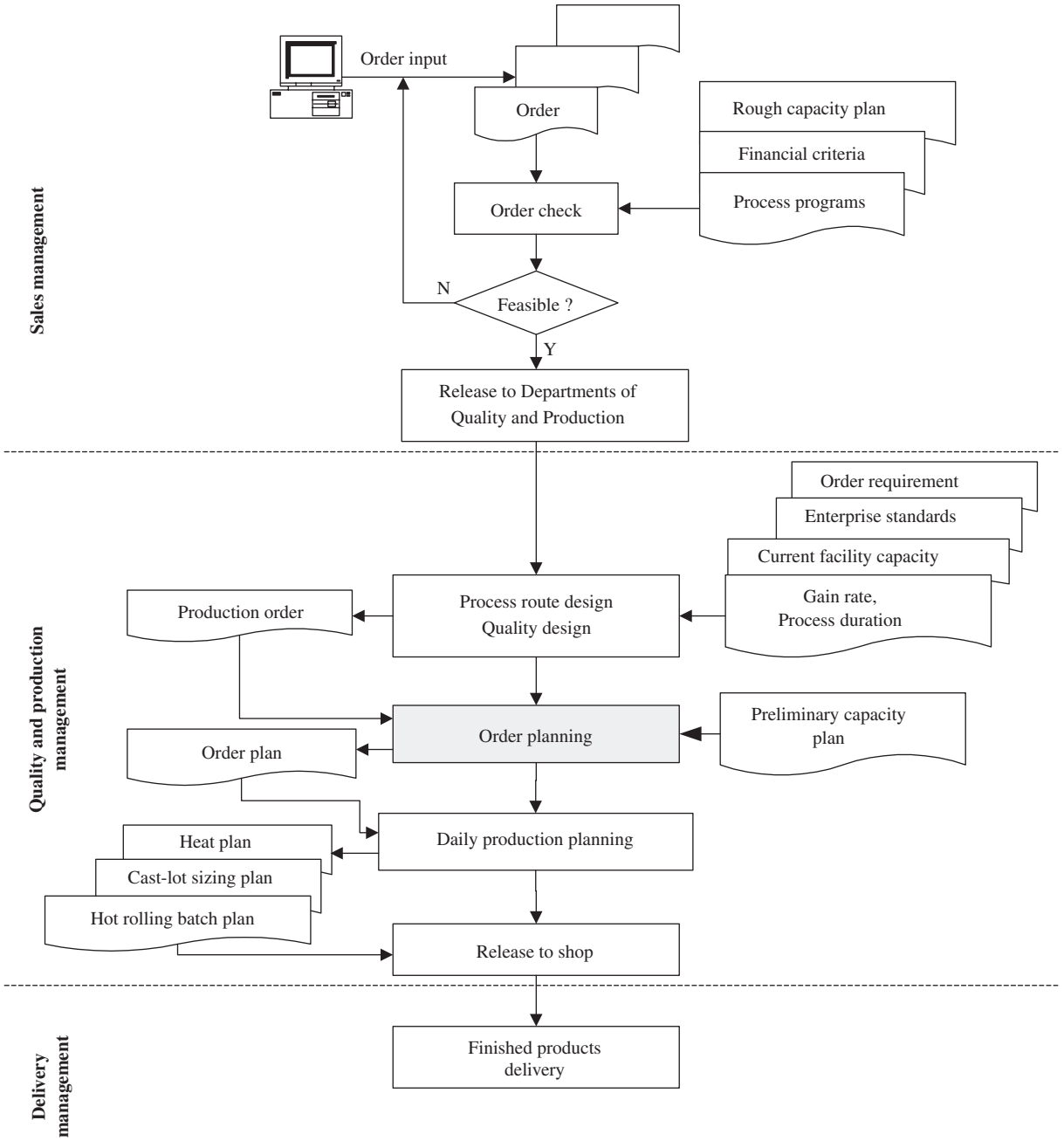

Fig. 2. Typical production organization process in steel plant.

of process routes and quality criteria design are illustrated in Fig. 1. After the above design processes, the *user orders* are then transformed into *production orders*, which specify manufacturing routes, quality criteria, material sizes, unit weight of WIP in every process through production line. The *production orders* could be classified into different categories according to their chemical ingredients, target width, target

gauge and hardness. After completing *production order* transformation, the order plan is made in production system to assign manufacturing time of orders in each production process.

2.3. Order planning

The *production order* guarantees that the user requirements are met for every piece of final product, but the production in iron–steel plants is a batch-manufacturing process. First, it involves gathering orders that have the same chemical ingredients for meeting the BOF capacity. It is called a "heat" for the necessary amount of molten steel to be poured into the BOF once. Usually, the molten steel is poured 5–7 times consecutively, and this sequence of heats is called a "cast". For instance, in a large-scale iron–steel plant in China, one slab weighs about 40 tons, one heat of molten steel weighs about 300 tons, therefore, it is necessary to gather orders together with the same chemical ingredients, similar width, gauge and due date, and compile them into the same heat plan and cast plan. Next, slabs cast in CC are rolled in HRM. To guarantee the product quality and reduce production cost, the slabs should be grouped and matched proportionally to rolling batch according to width, gauge, and hardness of target sheets when rolled in HRM (Tang et al., 2000b). Therefore, in the order-planning process both the production process program constraints and difficulties of making daily production plan should be taken into consideration, so as to guarantee that the output proportion of different category of orders is suitable in every manufacturing period.

The flow of order planning is given as follows: First, calculate the earliest and latest finish time of orders in every process through the production line basing on order due date and process duration table. The earliest finish time is calculated forward from plan date based on process duration table, while the latest finish time is calculated backward from due date based on process duration table. Second, select suitable finish time (in term of 5 days) of orders in every process from the interval between the earliest finish time and the latest finish time based on the production capacity constraints, inventory capacity constraints and output match proportion relation of different category of orders. Hence, the objectives of order planning are simultaneously to minimize total tardiness cost, balance utilization of production capacities and to minimize inventory cost of WIP and final products.

3. Model formulation

3.1. Order-planning model

The following notations are used in the model:

i Index of *production order*, $i = 1, \ldots, N$;
j Index of process, $j = 1, \ldots, J$;
T Planning horizon;
EF_{ij} The earliest finish time of order i in process j (in term of 5 days);
LF_{ij} The latest finish time of order i in process j (in term of 5 days);
D_i Due date of order i;
p_{ij} Duration of order i in process j;
$\tau_{ij}^{\min}$ The minimum time interval from finish in process $j-1$ to start in process j of order i;
$\tau_{ij}^{\max}$ The maximum time interval from finish in process $j-1$ to start in process j of order i;
α_i Tardiness cost per ton of order i;
β_j Penalty cost per ton if capacity utilization in process j diverges from target utilization;
γ_j Inventory cost per ton of WIP or final product in process j;

w_{ij} Weight in tons corresponding to order i at the output of process j;
E_{jt} The maximum production capacity of process j in period t;
$\overline{E}_{jt}$ Target capacity utilization of process j in period t;
I_{j0} Initial inventory level in process j;
$I_j^{\max}$ The maximum inventory capacity of process j;
$\varphi_{lkj}^{\min}$ The minimum output proportional of orders in category l to orders in category k in process j;
$\varphi_{lkj}^{\max}$ The maximum output proportional of orders in category l to orders in category k in process j;
Ω_l The set of all order categories, $l = 1, 2, \ldots, L$;
I_{jt} Inventory level of WIP and final products in process j in period t;

$$x_{ijt} = \begin{cases} 1 & \text{Process } j \text{ of order } i \text{ finishes in period } t, \quad i = 1, \ldots, N; \quad j = 1, \ldots, J; \\ 0 & \text{Others,} \qquad\qquad\qquad\qquad\quad t = 1, \ldots, T. \end{cases}$$

It is assumed that the process duration is limited to one period, namely, if process j of order i finishes in period t, it also starts in period t. Because the period unit is 5 days, the assumption is reasonable. Then the OPM is formulated as follows:

$$\text{Min } Z_1 = \sum_{i=1}^{N} \alpha_i \cdot w_{iJ} \cdot \max\left(\sum_{t=\text{EF}_{iJ}}^{\text{LF}_{iJ}} x_{iJt} \cdot t - D_i, 0 \right), \tag{1}$$

$$\text{Min } Z_2 = \sum_{j=1}^{J} \sum_{t=1}^{T} \beta_j \cdot \left| \sum_{i=1}^{N} w_{ij} \cdot x_{ijt} - \overline{E}_{jt} \right|, \tag{2}$$

$$\text{Min } Z_3 = \sum_{j=1}^{J} \sum_{t=1}^{T} \lambda_j \cdot I_{jt}, \tag{3}$$

s.t.

$$\sum_{t=\text{EF}_{ij}}^{\text{LF}_{ij}} x_{ijt} = 1, \quad i = 1, \ldots, N; \; j = 1, \ldots, J, \tag{4}$$

$$\sum_{i=1}^{N} w_{ij} \cdot x_{ijt} \leqslant E_{jt}, \quad j = 1, \ldots, J; \; t = 1, \ldots, T, \tag{5}$$

$$I_{j0} + \sum_{i=1}^{N} \left(\sum_{q=1}^{t} w_{ij} \cdot x_{ijq} - \sum_{q=1}^{t} w_{ij} \cdot x_{i(j+1)q} \right) \leqslant I_j^{\max}, \quad j = 1, \ldots, J; \; t = 1, \ldots, T, \tag{6}$$

$$\varphi_{lkj}^{\min} \leqslant \sum_{i \in \Omega_l} w_{ij} \cdot x_{ijt} \Big/ \sum_{i \in \Omega_k} w_{ij} \cdot x_{ijt} \leqslant \varphi_{lkj}^{\max}, \quad j = 1, \ldots, J; \; t = 1, \ldots, T; \; l, k = 1, \ldots, L, \tag{7}$$

$$\tau_{ij}^{\min} \leqslant \left(\sum_{t=\text{EF}_{ij}}^{\text{LF}_{ij}} t \cdot x_{ijt} - p_{ij} + 1 \right) - \sum_{t=\text{EF}_{ij-1}}^{\text{LF}_{ij-1}} t \cdot x_{i(j-1)t} \leqslant \tau_{ij}^{\max}, \quad i = 1, \ldots, N; \; j = 2, \ldots, J, \tag{8}$$

where objective function (1) is to minimize total tardiness cost, objective function (2) is to balance utilization of production capacities, objective function (3) is to minimize inventory cost of WIP and final

products. Eq. (4) ensures that orders are to be finished in every process only once. Eq. (5) guarantees that the capacity in every process is restricted at every period. Eq. (6) models the constraints imposed by inventory capacity of every process. Eq. (7) expresses that the output proportional relation of different category of orders is reasonable in every manufacturing period. Eq. (8) represents the technological precedence constraints between processes of every order. During the manufacturing processes, there are technological time lag constraints between some processes; for example, it has to be completed within one day from BOF to CC for every category of molten steel. However, for some category of steel, it should pass several days of cooling period from CC to HRM.

The OPM (1)–(9) is a multi-objective nonlinear 0–1 integer optimization problem. We could have considered multi-objective approaches. Nevertheless, it would have been very difficult to model the preferences of the various decision makers involved. In consequence, we model the preferences by weights for the various criteria, knowing that each optimal solution for a given weighted sum of criteria is a Pareto optimum for the whole multi-objective problem. Let ω_1, ω_2, and ω_3 be the important weights of the objectives, respectively, preferred by the planner. The objectives (1)–(3) are replaced by the following single objective:

$$\text{Min } Z = \omega_1 \cdot Z_1 + \omega_2 \cdot Z_2 + \omega_3 \cdot Z_3. \tag{10}$$

According to practical investigation in the large-scale iron–steel plant, the planning horizon is one month, namely includes 6 periods of 5 days, the production process consists of 4 major processes. Supposing that there are 100 pieces of orders that would be planned in a month, these orders could be classified into 4 categories, then there are at most $100 \times 4 \times 6 = 2400$ ($N = 100$, $J = 4$, $T = 6$) binary variables, $100 \times 4 + 4 \times 6 + 4 \times 6 + 4 \times 6 \times 6 \times 2 + 100 \times 3 \times 2 = 1336$ constraints in the model. Because this model is an integer programming problem with great size, it could not be solved in a reasonable computational time.

3.2. Pre-processing of order planning

For reducing the complexity of the solution process, the binary variables x_{ijt} are transformed in the following approach: $y_{ij} = \sum_{t=\mathrm{EF}_{ij}}^{\mathrm{LF}_{ij}} t \cdot x_{ijt}$, $\mathrm{EF}_{ij} \leqslant t \leqslant \mathrm{LF}_{ij}$. Then $\mathrm{EF}_{ij} \leqslant y_{ij} \leqslant \mathrm{LF}_{ij}$, y_{ij} and x_{ijt} have the following mapping relation:

$$x_{ijt} = \begin{cases} 1, & y_{ij} = t; \quad i = 1, \ldots, N; \quad j = 2, \ldots, J; \quad \mathrm{EF}_{ij} \leqslant t \leqslant \mathrm{LF}_{ij}, \\ 0, & \text{Others.} \end{cases} \tag{11}$$

For the same problem described above, after transformation the number of variables reduced to $100 \times 4 = 400$. However, with this transformation, some constraints are more difficult to write and probably need secondary variables to be introduced. In consequence, we do not formulate this problem with the integer programming model, but use the integer variables in our algorithm and transfer them back into binary format when dealing with constraints (5)–(7), provided that constraints (4) satisfy naturally. In the next section, a modified PSO algorithm is designed to solve it.

4. PSO Algorithm

4.1. Algorithm definition

PSO is an optimization technique proposed by Kennedy and Eberhart firstly in 1995 (Kennedy and Eberhart, 1995; Eberhart and Kennedy, 1995). Social behavior of organisms such as bird flocking and fish schooling motivated them to look into the effect of collaboration of species onto achieving

their goals as a group. Years of study for the dynamics of bird flocking resulted in the possibilities of utilizing this behavior as an optimization tool. In a PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution candidate, called a 'particle', flies in the problem search space looking for the optimal position to land (similar to the search process for food of a bird swarm). The performance of each particle is measured according to a predefined fitness function, which is usually proportional to the objective function. A particle, as time passes through his quest, adjusts its position according to its own 'experience', as well as according to the experience of neighboring particles. Tracking and memorizing the best position encountered build particle's experience. For that reason, the PSO algorithm possesses a memory (i.e. every particle remembers the best position it reached during the past). PSO system combines local search methods through self-experience with global search methods through neighboring experience, with an attempt to balance exploration and exploitation.

Assuming that the search space is D-dimensional, the ith particle of the swarm is represented by the D-dimensional vector $Y_i = (y_{i1}, y_{i2}, \ldots, y_{iD})$ and the best particle of the swarm, i.e. the particle which found the smallest function value (where the optimization problem is to be minimized) heretofore is denoted by index g. The best previous position (i.e. the position giving the lowest function value) of the ith particle is recorded and represented as $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and the position change (velocity) of the ith particle is $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The particles are manipulated according to the following equations (the superscripts denote the iteration):

$$V_i^{n+1} = w \cdot V_i^n + c_1 \cdot r_{i1}^n \cdot (P_i^n - Y_i^n) + c_2 \cdot r_{i2}^n \cdot (P_g^n - Y_i^n), \tag{12}$$

$$Y_i^{n+1} = Y_i^n + V_i^{n+1}, \tag{13}$$

where $i = 1, 2, \ldots, M$; M is the swarm's size, n is iteration index, and w is the *inertia weight*. Experimental results suggest that it is better to set the *inertia weight* to a large value initially to promote global exploration of the search space, and then decrease it gradually to obtain refined solutions at the final stage of the search (Eberhart and Shi, 2001). c_1 and c_2 are two positive constants, called the *cognitive* and *social* parameter, respectively. Suitable selection of w, c_1 and c_2 could provide a balance between the global and the local search. r_{i1}^n and r_{i2}^n are two random numbers uniformly distributed within the range [0,1].

Eq. (12) is used to calculate the particle's new velocity according to its previous velocity and the distances of its current position from its own best historical position and the best found position by any particle. Then the particle flies toward a new position according to Eq. (13). This process is repeated until user-defined stopping criteria are satisfied.

4.2. The modified PSO

PSO was initially developed for solving optimization problem with real-valued variables and without constraints, such as evolving the neural network weights, function optimization, etc. However, in this paper the model has inequality constraints, and its variables are integer-valued. So a modified PSO is designed to solve it.

Observing the OPM, one could find that constraints (4) are met naturally after its transformation as in (11), constraints (5), (6) and (7) are loose in practical problem, and constraints (8) are strict because they reflect requirements of production process programs. Though each process of each order has a feasible interval $[EF_{ij}, LF_{ij}]$ for the finish time, the intervals are possibly overlapped in some orders. Because of the constraints of Eq. (8), if a finish time is assigned to a process of an order, the feasible interval $[EF_{ij}, LF_{ij}]$ of its preceding and successive processes may change. What is more, the unsuitable assignment may result that no feasible assignments could be found.

Based on the above analysis, we deal with the constraints in the following way: for constraints (5), (6) and (7), penalty functions are imposed to incorporate these constraints into the objective function (10), where χ_1, χ_2 and χ_3 are penalty coefficients of constraints (5), (6) and (7), respectively,

$$\begin{aligned} P = & \sum_{j=1}^{J}\sum_{t=1}^{T}\chi_1 \cdot \max\left\{\left(\sum_{i=1}^{N} w_{ij}\cdot x_{ijt} - E_{jt}\right), 0\right\} \\ & + \sum_{j=1}^{J}\sum_{t=1}^{T}\chi_2 \cdot \max\left\{\left(I_{j0} + \sum_{i=1}^{N}\left(\sum_{q=1}^{t} w_{ij}\cdot x_{ijq} - \sum_{q=1}^{t} w_{ij}\cdot x_{i(j+1)q}\right) - I_{jt}\right), 0\right\} \\ & + \sum_{j=1}^{J}\sum_{t=1}^{T}\chi_3 \cdot \max\left\{\left(\varphi_{lkj}^{\min}\cdot\sum_{i\in\Omega_k} w_{ij}\cdot x_{ijt} - \sum_{i\in\Omega_l} w_{ij}\cdot x_{ijt}\right), \left(\sum_{i\in\Omega_l} w_{ij}\cdot x_{ijt} - \varphi_{lkj}^{\max}\cdot\sum_{i\in\Omega_k} w_{ij}\cdot x_{ijt}\right), 0\right\}. \end{aligned} \tag{14}$$

Then, the original objective functions are transformed into the following format:

$$\text{Min } Z' = \omega_1 \cdot Z_1 + \omega_2 \cdot Z_2 + \omega_3 \cdot Z_3 + P. \tag{15}$$

For constraints (8), infeasible particles are eliminated during the process of initial particle swarm generation and iteration. In the process of generating initial particle swarm, the finish time y_{ij} of order i in process j is assigned as follows: a normally distributed number r is generated in the way of $r \sim N((\mathrm{EF}_{ij} + (\mathrm{LF}_{ij} - \mathrm{EF}_{ij})/2), (\mathrm{LF}_{ij} - \mathrm{EF}_{ij})/2)$, set y_{ij} to be the closest integer value of r in $[\mathrm{EF}_{ij}, \mathrm{LF}_{ij}]$. If a particle is infeasible for constraints (8) for some orders, regenerate the corresponding part again. Repeat the above process until M feasible particles are generated, where M is the swarm size.

The PSO flow is illustrated in Procedure 1, where $\mathrm{Vel}_{\max}$ is the maximum velocity of particle in each direction, w_0 is the initial inertia weight, maxIter is the maximum number of iterations, the other notations are defined as in previous sections.

Procedure 1

Step 1: Initialize $\mathrm{Vel}_{\max}$, w_0, $g = 1$, $c_1 = c_2 = 2.0$, M, maxIter;
Step 2: Generate an array of feasible particles with random positions and velocities on D-dimensions;
Step 3: Calculate function value of every particle with (15); Initialize P_i with current particle;
Step 4: For $n = 1$ To maxIter

For $i = 1$ To M
$V_i^{n+1} = w \cdot V_i^n + c_1 \cdot r_{i1}^n \cdot (P_i^n - Y_i^n) + c_2 \cdot r_{i2}^n \cdot (P_g^n - Y_i^n)$;
If $v_{id} > \mathrm{Vel}_{\max}(d = 1, D)$, Then $v_{id} = \mathrm{Vel}_{\max}$;
If $v_{id} < -\mathrm{Vel}_{\max}(d = 1, D)$, Then $v_{id} = -\mathrm{Vel}_{\max}$;
End For
For $i = 1$ To M
$Y_i^{n+1} = Y_i^n + V_i^{n+1}$;
Set $y_{ij}, j = 1, \ldots, J$ to be the closest integer in $[\mathrm{EF}_{ij}, \mathrm{LF}_{ij}]$;
Judge feasibility of every particle with (8), substitute new particles for infeasible particles;
Calculate function value of every particle using (15);
End For
Update P_i^n and P_g^n;
$w = w_0 - (w_0 - 0.4) \times n/\mathrm{MaxIter}$;
End for
Step 5: Output P_g^n

5. Computational results

To test the effectiveness and efficiency of the proposed model and algorithm, three actual instances from a large-scale iron–steel plant in China are cited in this section. These three instances termed as plan1, plan2 and plan3, respectively, are actual order plans previously produced in the plant, and consist of 51, 138 and 249 individual orders, respectively. The algorithm has been coded in Java language under Windows 2000 system and run on a Pentium III/1G/128M personal computer.

In this study, the particle swarm size M, maximum iteration number maxIter, maximum particle velocity $\mathrm{Vel}_{\max}$ are set as $M = 500$, maxIter = 2000, $\mathrm{Vel}_{\max} = 2$, respectively. The initial *inertia weight* is set as $w_0 = 1.0$, and w reduces linearly to 0.4; $c_1 = c_2 = 2.0$. The penalty coefficients of constraints (5), (6) and (7) are set as $\chi_1 = \chi_2 = \chi_3 = 10$. In initial experiments phase, these parameters are set to be other different values, whereas the PSO is quite robust with them, except that particle swarm size M has little more effect on the algorithm. As a whole, these parameter values provide good solutions in computational tests with $M = 500$. Computation experiments have shown that by increasing M, no significant improvement could be observed. To study the impact of different weights preferred by the decision maker on the objective function values, several sets of weight combinations are used for each instance in the experiments.

Fig. 3 illustrates a typical run process with $M = 500$, $\omega_1 = 0.5$, $\omega_2 = 0.3$, and $\omega_3 = 0.2$ for the order-planning instance that consists of 51 individual orders. In Fig. 3, the top curve describes the evolving process of the average objective function value calculated as Eq. (15) of the swarm, the bottom curve describes the evolving process of the global best objective function value in the run. In the initial search period, because of the penalty of constraints (5), (6) and (7) calculated as Eq. (14), the objective function value was very high; with the evolving process, the penalty diminished and the plan became more suitable, and hence the objective function value became minimum. The figure shows that the algorithm converged within 1400 iterations. In fact, this is the general result for all instances and parameter settings in the experiments.

To test the effectiveness of the proposed PSO algorithm, NB = 10,000,000, $5 \times$ NB, $10 \times$ NB, $50 \times$ NB and $100 \times$ NB solutions are randomly generated based on heuristic rules for comparison. The main ideas of the rules-based random algorithm are similar to the process of initial particle swarm generation. In our experiments, we designed a backward and forward procedure to determine variables y_{ij}, so that constraints (8) are satisfied. In the backward process, the variables y_{iJ} $(i = 1, \ldots, N)$ are randomly specified first, and the variables y_{ij} $(i = 1, \ldots, N;\ j = J-1, \ldots, 1)$ are determined stage-by-stage taking into account the already determined variables. If no feasible solution could be generated after the backward process, the forward process would be called. If no feasible solution could be generated yet after the backward and forward procedure, a new loop restarts. This procedure would be iterated until a feasible solution is generated. The rules-based random algorithm for generating a feasible solution is defined in detail in Appendix A.

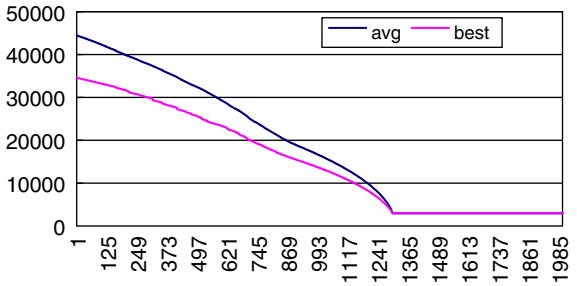

Fig. 3. Typical run process for a problem instance.

Table 1
The results of different order-planning approaches

	Total orders	Output (ton)	Plans by a senior planner			Weights (ω_i)			Plans generated randomly			Plans by PSO algorithm			PSO CPU(S)
			Z_1	Z_2	Z_3	ω_1	ω_2	ω_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3	
Plan1	51	40,620	4560	3231	1215	0.5	0.3	0.2	5072	3877	1290	3649	2907	1283	557.57
						0.5	0.2	0.3	5103	3968	1265	3712	3015	1246	
						0.3	0.5	0.2	5237	3612	1257	4039	2874	1295	
						0.3	0.2	0.5	5125	3912	1223	3981	2958	1202	
						0.2	0.5	0.3	5312	3596	1247	4275	2740	1301	
						0.2	0.3	0.5	5297	3901	1245	4318	3032	1198	
Plan2	138	108,715	11,023	8528	2783	0.5	0.3	0.2	12,327	9807	3061	8267	6822	2366	1578.74
						0.5	0.2	0.3	12,439	9923	3012	8301	6973	2212	
						0.3	0.5	0.2	14,260	8956	3105	9985	6312	2401	
						0.3	0.2	0.5	14,109	10,020	2938	10,130	7018	2137	
						0.2	0.5	0.3	15,501	8876	3115	10,245	6217	2364	
						0.2	0.3	0.5	15,039	9765	2894	11,005	7068	2028	
Plan3	249	183,748	19,472	14,728	4697	0.5	0.3	0.2	22,392	16,200	5401	13,630	11,489	3757	2812.25
						0.5	0.2	0.3	23,021	16,793	5219	13,598	12,973	3654	
						0.3	0.5	0.2	25,473	14,839	5512	14,589	10,369	3802	
						0.3	0.2	0.5	24,978	16,540	4973	14,960	13,063	3027	
						0.2	0.5	0.3	27,351	14,532	5156	15,046	10,291	3593	
						0.2	0.3	0.5	26,793	15,930	5020	14,985	12,039	3321	

The results of PSO algorithm, randomly generated plans and the original plans done by a senior planner through human–machine coordination are summarized in Table 1. Columns 4–6, 10–12 and 13–15 of Table 1 give the three objective function values calculated as Eqs. (1), (2) and (3) obtained by a senior planner, randomly generated and the PSO algorithm, respectively. Columns 7–9 in Table 1 are given the six sets of weight combinations of the three objectives. For the PSO algorithm, the best solutions of 10 runs are picked under each set of weight combinations; for randomly generated plans, the best NB solution is picked under each set of weight combinations. Column 16 is the average CPU time per PSO run in seconds. The best solutions picked up when NB, $5 \times$ NB, $10 \times$ NB, $50 \times$ NB and $100 \times$ NB solutions were randomly generated are summarized in Table 2.

From computational results one could observe that the performance of the model and algorithm in this paper outperforms that made by a senior planner, and this is randomly generated. Different sets of weight combinations preferred by the decision maker on the objective function values may conduce different optimization preference of the algorithm so as to provide multiple choices for the decision maker. Although the total numbers of solutions are same, namely 10,000,000, the solutions obtained by PSO are superior to that randomly generated in average, which validates the performance of PSO algorithms.

6. Conclusions

Reducing production cost, improving economic benefit and providing better services for consumers by effectively using current resources are an important subject that modern enterprises have been facing. This pressure could be met, at least in part, by making more reasonable and efficient production plans. Order planning is an important production management job to improve business performance by ameliorating

Table 2
The best solutions picked up when different number of solutions were randomly generated

	Weights (ω_i)			NB			$5 \times$ NB			$10 \times$ NB			$50 \times$ NB			$100 \times$ NB		
	ω_1	ω_2	ω_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3
Plan1	0.5	0.3	0.2	5072	3877	1290	4180	3626	1272	4094	3575	1255	4022	3316	1246	3906	3246	1222
	0.5	0.2	0.3	5103	3968	1265	4849	3384	1255	4645	3298	1245	4194	3010	1216	3765	2889	1199
	0.3	0.5	0.2	5237	3612	1257	5004	3441	1255	4825	3027	1252	4794	3023	1249	3913	2774	1233
	0.3	0.2	0.5	5125	3912	1223	4851	3874	1215	4745	3734	1194	4458	3347	1170	3830	3239	1159
	0.2	0.5	0.3	5312	3596	1247	4865	3594	1245	4566	3055	1240	4439	3008	1235	4062	2859	1235
	0.2	0.3	0.5	5297	3901	1245	5279	3592	1245	5184	3440	1232	4446	3259	1222	4261	3092	1186
Plan2	0.5	0.3	0.2	12,327	9807	3061	11,882	8107	3047	11,335	7940	2582	10,772	7535	2499	10,068	7099	2471
	0.5	0.2	0.3	12,439	9923	3012	12,242	9884	3008	11,509	9383	2940	9299	8451	2738	8724	8212	2579
	0.3	0.5	0.2	14,260	8956	3105	13,178	8795	2975	12,744	8350	2923	11,139	7353	2745	9627	7351	2638
	0.3	0.2	0.5	14,109	10,020	2938	13,848	9123	2834	11,111	8923	2731	9725	8891	2685	9712	6757	2437
	0.2	0.5	0.3	15,501	8876	3115	14,038	6740	2917	11,050	6242	2876	10,195	6190	2829	9952	6114	2416
	0.2	0.3	0.5	15,039	9765	2894	13,809	8896	2740	12,928	8749	2697	12,316	8402	2526	12,153	7113	2203
Plan3	0.5	0.3	0.2	22,392	16,200	5401	21,287	15,823	4989	18,852	14812	4811	18,494	12,783	4197	15,489	11,802	3585
	0.5	0.2	0.3	23,021	16,793	5219	22,388	15,821	4405	20,617	14712	4037	19,341	14,274	3909	16,561	13,884	3722
	0.3	0.5	0.2	25,473	14,839	5512	23,096	11,905	5134	20,337	11874	4814	17,755	11,343	4776	16,543	10,226	4303
	0.3	0.2	0.5	24,978	16,540	4973	24,025	16,066	4855	22,333	15743	4426	18,427	14,996	3889	14,229	13,524	3817
	0.2	0.5	0.3	27,351	14,532	5156	24,528	13,856	4788	23,127	13362	4477	18,829	10,497	4292	15,737	10,301	3926
	0.2	0.3	0.5	26,793	15,930	5020	21,373	13,649	4937	17,085	13465	4236	15,626	12,735	4138	15,204	12,355	4124

customer services, maximizing plant throughputs and reducing inventory levels in iron–steel plants. In this paper, based on the investigation of the order-planning strategies and optimization objectives in steel plant, a multi-objective OPM is formulated. Specific PSO algorithm is designed to solve the model. Computation results show that the model and algorithm are superior to human–machine coordination approach. Experiments reveal that PSO algorithms are effective for solving this kind of problems. Future works include embedding senior planner's know-how into the algorithms so as to further improve the effectiveness and efficiency of the algorithm.

Acknowledgements

This research is financially supported by National Natural Science Foundation of China through Approval No. 70301007, 70002009, National 863/CIMS Scheme of China through approval No. 2002AA412010, and Natural Sciences Foundation of Liaoning province in China through Approval No. 20021011. We also want to thank two anonymous referees on their constructive comments for this paper.

Appendix A. The rules-based random algorithm for generating a random solution

After the transformation defined as formula (11), constraints (8) could be formulated as follows:

$$\tau_{ij}^{\min} \leqslant (y_{ij} - p_{ij} + 1) - y_{i(j-1)} \leqslant \tau_{ij}^{\max}, \quad i = 1, \ldots, N; \ j = 2, \ldots, J. \tag{A.1}$$

Considering $y_{ij} \in [\mathrm{EF}_{ij}, \mathrm{LF}_{ij}]$, the following constraints on y_{ij} should be satisfied:

$$y_{i(j-1)} \leqslant \min\{\mathrm{LF}_{i(j-1)}, (y_{ij} - p_{ij} + 1 - \tau_{ij}^{\min})\}, \quad i = 1, \ldots, ; N; \ j = 2, \ldots, J, \tag{A.2}$$

$$y_{i(j-1)} \geqslant \max\{\mathrm{EF}_{i(j-1)}, (y_{ij} - p_{ij} + 1 - \tau_{ij}^{\max})\}, \quad i = 1, \ldots, N; \ j = 2, \ldots, J, \tag{A.3}$$

$$y_{ij} \leqslant \min\{\mathrm{LF}_{ij}, (y_{i(j-1)} + p_{ij} - 1 + \tau_{ij}^{\max})\}, \quad i = 1, \ldots, N; \ j = 2, \ldots, J, \tag{A.4}$$

$$y_{ij} \geqslant \max\{\mathrm{EF}_{ij}, (y_{i(j-1)} + p_{ij} - 1 + \tau_{ij}^{\min})\}, \quad i = 1, \ldots, N; \ j = 2, \ldots, J. \tag{A.5}$$

Based on constraints (A.2)–(A.5), we designed a backward and forward procedure to determine variables y_{ij} so that constraints (8) are satisfied. In the backward process, the variables y_{iJ} $(i = 1, \ldots, N)$ are randomly specified first, and the variables y_{ij} $(i = 1, \ldots, N; j = J - 1, \ldots, 1)$ are determined stage by stage taking into account the already determined variables based on constraints (A.2)–(A.5). If no feasible solution could be generated after the backward process, the forward process would be called to generate a feasible solution. If no feasible solution could be generated yet after the backward and forward procedure, a new loop restarts. This procedure would be iterated until a feasible solution is generated. The rules-based random algorithm for generating a feasible solution is defined in detail as follows.

The rules-based random algorithm:

For $i = 1$ To N
FeasibleFlag = 0
  Do
    //The backward process
    Generate a normally distributed number $r \sim N((\mathrm{EF}_{iJ} + (\mathrm{LF}_{iJ} - \mathrm{EF}_{iJ})/2), (\mathrm{LF}_{iJ} - \mathrm{EF}_{iJ})/2)$ set y_{iJ}, to be the closest integer value of r in $[\mathrm{EF}_{iJ}, \mathrm{LF}_{iJ}]$
    For $j = J-1$ To 1
      $a \coloneqq \max\{\mathrm{EF}_{ij}, (y_{i(j+1)} - p_{i(j+1)} + 1 - \tau_{i(j+1)}^{\max})\}$
      $b \coloneqq \min\{\mathrm{LF}_{ij}, (y_{i(j+1)} - p_{i(j+1)} + 1 - \tau_{i(j+1)}^{\min})\}$
      If $a > b$ Then break this inner loop
      Generate a normally distributed number $r \sim N((a + (b - a)/2), (b - a)/2)$, set y_{ij} to be the closest integer value of r in $[a, b]$
      If $j = 1$ Then FeasibleFlag = 1
    End for
    If (FeasibleFlag = 0) Then
    {//The forward process
      Generate a normally distributed number $r \sim N((\mathrm{EF}_{i1} + (\mathrm{LF}_{i1} - \mathrm{EF}_{i1})/2), (\mathrm{LF}_{i1} - \mathrm{EF}_{i1})/2)$, set y_{i1} to be the closest integer value of r in $[\mathrm{EF}_{i1}, \mathrm{LF}_{i1}]$
      For $j = 2$ To J
        $a \coloneqq \max\{\mathrm{EF}_{ij}, (y_{i(j-1)} + p_{ij} - 1 + \tau_{ij}^{\min})\}$
        $b \coloneqq \min\{\mathrm{LF}_{ij}, (y_{i(j-1)} + p_{ij} - 1 + \tau_{ij}^{\max})\}$
        If $a > b$ Then break this inner loop
        Generate a normally distributed number $r \sim N\big((a + (b - a)/2), (b - a)/2\big)$, set
        y_{ij} to be the closest integer value of r in $[a, b]$
        If $j = J$ Then FeasibleFlag = 1
      End for
    }

While (FeasibleFlag = 0)
End For
Output the feasible solution for constraints (8) $(y_{i1}, \ldots, y_{iJ})$

Considering the rules-based random algorithm is similar to the process of initial particle swarm generation, and the number of total feasible solutions generated in solving the process is identical. The differences are that the randomly generated solutions are independent of each other, but in PSO algorithm the solutions evolve in a predefined mechanism. Therefore, comparison results of the two approaches could verify the effectiveness of the evolution mechanism defined in PSO Algorithm.

References

Cohen, J., Wallmeier, H.M., Twisselmann, U., Lantz, B., O'Dell, D., 1997. Enhancing hot mill slab yield, usage and throughput using expert systems and numerical optimization. Proceedings of Ironmaking Conference, Chicago, IL, USA, pp. 663–672.

Eberhart, R.C., Kennedy, J., 1995. A new optimizer using particle swarm theory, Proceedings of the Sixth International Symposium on Micro Machine and Human Science, IEEE Service Center, Piscataway, NJ, pp. 39–43.

Eberhart, R.C., Shi, Y., 2001. Particle swarm optimization: Developments, applications and resources. Proceedings of Congress on Evolutionary Computation 2001, IEEE Service Center, Piscataway, NJ, pp. 81–86.

Fujii, N., Kumamoto, K., Asada, K., 1996. Development of consistent scheduling system for sheet metal production. Proceedings of the Japan/USA Symposium on Flexible Automation, vol. 2, pp. 1423–1426.

Kennedy, J., Eberhart, R.C., 1995. Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, vol. IV, IEEE Service Center, Piscataway, NJ, pp. 1942–1948.

Portmann, M.C., Rohr, D., 1995. Hierarchical production management applied to an iron and steel industry. Journal of Intelligent Manufacturing 6, 79–85.

Redwine, C.N., Wismer, D.A., 1974. A mixed integer programming model for scheduling orders in a steel mill. Journal of Optimization Theory and Applications 14, 305–318.

Sasidhar, B., Achary, K.K., 1991. Multiple arc network model of production planning in a steel mill. International Journal of Production Economics 22, 195–202.

Tamura, R., Nagai, M., Nakagawa, Y., TanizakiI, T., Nakajima, H., 1998. Synchronized Scheduling Method in Manufacturing Steel Sheets. International Transactions in Operational Research 5, 189–199.

Tang, L., Liu, J., Rong, A., Yang, Z., 2000a. A mathematical programming model for scheduling steelmaking-continuous casting production. European Journal of Operational Research 120, 423–435.

Tang, L., Liu, J., Rong, A., Yang, Z., 2000b. A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron & Steel Complex. European Journal of Operational Research 124, 267–282.

Zhang, T., Wang, M., Tang, L., Song, J., Yang, J., 2000. Method for order planning of the steel plant based on the MTO management system. Chinese Journal of Control and Decision 15, 649–653.