Int j simul model 19 (2020) 4, 678-688
ISSN 1726-4529 Original scientific paper

MULTI-OBJECTIVE MASTER PRODUCTION SCHEDULE FOR BALANCED PRODUCTION OF MANUFACTURERS

Wang, C.[*,#]; Yang, B.[*] & Wang, H. Q.[**]
[*] Qiqihar University, Qiqhar 161006, China
[**] Jiangxi University of Finance and Economics, Nanchang 330013, China
E-Mail: wangcheng@qqhru.edu.cn ([#] Corresponding author)

Abstract
Focusing on the balanced use of production capacity in the formulation of master production schedule (MPS), this paper sets up a single-product, multi-stage, multi-objective MPS model based on balanced production. Whereas the model aims to achieve multiple objectives through nonlinear integer programming, a genetic algorithm based on automatic transformation (AT-GA) was designed to solve the model. Specifically, the chromosomes were encoded as integers to satisfy the model constraints; the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) was adopted to handle the four nonlinear objectives of the model, thereby obtaining the fitness function; the fuzzy logic control (FLC) was introduced to automatically adjust the crossover and mutation parameters, and balance the global and local search abilities of the GA, enhancing the computing power of the algorithm. The experimental results show that the AT-GA can effectively solve the multi-objective MPS optimization problem under balanced production.
(Received in July 2020, accepted in October 2020. This paper was with the authors 1 month for 1 revision.)

Key Words: Manufacturer, Master Production Schedule (MPS), Balanced Production, Multiple Objectives

1. INTRODUCTION

Modern manufacturers often implement unified planning and management of resources, with the aid of enterprise resource planning (ERP) system: Based on the demand schedule, the master production schedule (MPS) is prepared to balance and coordinate production demand with available capacity, and to drive the execution of the demand schedule. As the crux of the ERP system, the MPS bridges market demand with product manufacturing, and provides an effective tool to schedule the production process.

The traditional ways to formulate the MPS have two main defects: First, without considering the constraints of production capacity, the production schedule needs to be adjusted repeatedly against the production capacity; no optimization strategy is available to fully resolve any serious conflict in production capacity. Second, the production capacity cannot be utilized in a balanced manner. Some optimization strategies treat production capacity as the constraint, and work to optimize objectives like production cost, and consumer service, but ignore the balanced use and optimization of production capacity. The traditional methods cannot adapt well to the production environment with unbalanced distribution of production capacity and high requirements on production management.

There is not yet a universal and mature model or algorithm for MPS optimization under balanced production. Some scholars attempted to make up for the defects of the demand schedule [1-4], and some suggested optimizing the MPS and demand schedule at the same time [5, 6]. On the balanced use of production capacity, Liu et al. [7] developed the first standardization framework for the typical MPS of balanced production in China's aviation manufacturing. To generate a balanced MPS for automakers, Li and Qin [8] presented a mixed integer programming (MIP) model, provided the constructive solution algorithm, and proved through case analysis that the model and algorithm can converge to high-quality balanced solution. Xu et al. [9] proposed the equilibrium optimization of the MPS considering bottleneck capacity and gave the principle and algorithm of heuristic solution; under the constraints of product demand and delivery time, their approach optimizes inventory and production cost, and realizes balanced distribution of bottleneck capacity on the planning horizon, making the production schedule more stable and feasible. Based on the theory of optimal constraints, Xu et al. [10] studied the progress and capacity balance of Deli Molded Products Co., Ltd., and demonstrated the rationality of the company's production schedule by analysing the relevant data and computing the equipment utilization rate.

From different angles, the above studies provide the optimization paths and methods for the balanced use of production capacity, in view of the varied features of different enterprises. However, the balanced utilization of production capacity is mainly achieved by heuristic algorithms with production capacity as the constraint, rather than models or algorithms with production capacity as the optimization objective.

In actual production, production capacity both serves as a necessary constraint of production operation, and an optimization objective of production management. In modern supply chain systems, there exist many small and medium-sized enterprises that provide supporting products to large manufacturers. Lacking diverse products and demand sources, these enterprises have special demand for the balanced use of production capacity.

Drawing on the relevant literature [11, 12], this paper puts forward a single-product, multi-stage, multi-objective MPS model based on balanced production, which integrates production schedule with production capacity control. Under the premise of meeting the market demand in each stage, the proposed model aims to prepare the MPS that achieves multiple objectives of performance management, including balanced production, on the planning horizon. Apart from constraints like net demand and production capacity, the MPS was formulated with balanced production as a production optimization objective. In addition, the proposed model also considers three other objectives: just-in-time (JIT) delivery, inventory occupation, and overtime production.

The remainder of this paper is organized as follows: Section 2 establishes a nonlinear integer programming model according to the features of the problem, including the parameters, objectives, and constraints; Section 3 proposes a genetic algorithm based on automatic transformation (AT-GA), in the light of the features of the established model; Section 4 verifies the ability of AT-GA to handle multi-objective optimization problem and its search ability through contrastive experiments; Section 5 puts forward the conclusions.

2. MODEL CONSTRUCTION

2.1 Parameters and hypotheses

(1) *Subscripts and thresholds*

t – planning period,

T – planning horizon, i.e., the sum of planning periods.

(2) *Parameters*

FI – beginning inventory of planning horizon,

UQ_t – number of products that can be manufactured through normal production in period t,

NQ_t – number of products that can be manufactured through overtime production in period t,

L – penalty coefficient for a unit number of products per unit period falling short of demand,

J – cost coefficient for unbalanced production in planning horizon,

K – cost coefficient of inventory occupation for a unit number of products per unit period,

A – cost coefficient of overtime pay for producing a unit number of products per unit period under overtime production,

GD_t – product demand in period *t*.

(3) *Variables*

PQ_t – planned number of products in period *t*,
LQ_t – cumulative number of products falling short of demand in period *t*,
TQ_t – total number of products that can be manufactured in period *t*,
BI_t – beginning inventory in period *t*,
EI_t – ending inventory in period *t*,
KI_t – mean inventory in period *t*,
JQ – standard deviation of the number of products manufactured in planning horizon,
AQ_t – number of products manufactured under overtime production in period *t*,
LC – total penalty for products falling short of demand in planning horizon,
JC – total penalty for unbalanced production in planning horizon,
KC – total cost of inventory occupation in planning horizon,
AC – total overtime pay in planning horizon.

(4) *Hypothesis*

H1. Throughout the planning horizon *T*, the manufacturer only manufactures one kind of products from a single resource to satisfy market demand; the production capacity can be characterized by the number of products.

H2. During each planning period, the manufacture can accurately determine the market demand.

H3. Depending on the market demand, the manufacturer can enhance its production capacity through a certain amount of overtime production, in addition to the normal production.

H4. Overproduction is not allowed, that is, the market demand should not be smaller than the sum of the number of products in the planning horizon and the beginning inventory of the planning horizon.

H5. The manufacturer needs to pay penalties, if it fails to meet the market demand in the current period, due to the lack of production capacity or inventory.

H6. The stability of production schedule in each period is an appraisal index of manufacturer performance; the manufacturer needs to pay an additional penalty if the production in planning period(s) is not balanced.

2.2 Objectives

Through the above analysis, four relatively independent objectives were selected for the multi-objective integrated MPS model: total penalty for products falling short of demand in planning horizon *LC*, total penalty for unbalanced production in planning horizon *JC*, total cost of inventory occupation in planning horizon *KC*, and total overtime pay in planning horizon *AC*. The four objectives should all be minimized:

$$Ob_1 = \min\{LC\} \tag{1}$$

$$Ob_2 = \min\{JC\} \tag{2}$$

$$Ob_3 = \min\{KC\} \tag{3}$$

$$Ob_4 = \min\{AC\} \tag{4}$$

2.3 Constraints

$$BI_t = \begin{cases} FI & \forall (t = 1) \\ EI_{(t-1)} & \forall (t > 1) \end{cases} \tag{5}$$

$$LQ_t = \begin{cases} \text{Max}\,(0,\ GD_1 - PQ_1 - BI_t) & \forall(t=1) \\ \text{Max}\,(0,\ LQ_{t-1} + GD_t - PQ_t - BI_t) & \forall(t>1) \end{cases} \tag{6}$$

$$EI_t = \begin{cases} \text{Max}\,(0,\ BI_1 + PQ_1 - GD_1) & \forall(t=1) \\ \text{Max}\,(0,\ BI_t + PQ_t - GD_t - LQ_{t-1}) & \forall(t>1) \end{cases} \tag{7}$$

$$KI_t = \frac{(BI_t + EI_t)}{2} \tag{8}$$

$$TQ_t = NQ_t + UQ_t \tag{9}$$

$$LC = \sum_{t=1}^{T} (LQ_t) \times L \tag{10}$$

$$JQ = \sqrt{\frac{\sum_{t=1}^{T} (PQ_t - \overline{Q})^2}{T-1}} \quad \text{where, } \overline{Q} = \frac{\left(\sum_{t=1}^{T} PQ_t\right)}{T} \tag{11}$$

$$JC = JQ \times J \tag{12}$$

$$KC = \sum_{t=1}^{T} (KI_t) \times K \tag{13}$$

$$AQ_t = \text{Max}\left\{PQ_t - UQ_t, 0\right\} \tag{14}$$

$$AC = \sum_{t=1}^{T} (AQ_t) \times A \tag{15}$$

$$0 \le PQ_t \le TQ_t \quad PQ_t \in Z \tag{16}$$

$$\sum_{t=1}^{T} (PQ_t) = \sum_{t=1}^{T} (GD_t) - FI \tag{17}$$

3. AT-GA

3.1 Encoding and initialization of chromosomes

(1) Encoding

The chromosomes were coded as integers. In the population, each chromosome represents a decision variable (PQ_t, $t = 0, 1, 2, ..., T$), which is an integer variable. In the AT-GA, each chromosome has $1 \times T$ dimensions:

$$\begin{aligned} Chr_l\ (\tau) &= [chr_l^1(\tau), ..., chr_l^t(\tau), ..., chr_l^T(\tau)] \\ &= [PQ_1, ..., PQ_t, ..., PQ_T] \end{aligned} \tag{18}$$

where, τ is the number of iterations, $\tau = 1, 2, ..., I$ (I is the maximum number of iterations); l is the serial number of chromosomes, $l = 1, 2, ..., L$; L is the population size. Thus, the position of each chromosome is coded as a $1 \times T$-dimensional vector to represent a point in the solution space.

(2) Initialization

To create feasible chromosomes under model constraints, each chromosome was initialized as follows:

Step 1: Let $t = 1$ and $m = 0$ (initialize gene t in the chromosome).

Step 2: Let $p = TQ_t$ and $q = \sum_{t=1}^{T} GD_t - FI - m$ (determine the upper bound of the chromosome).

Step 3: If $q < 0$, return to Step 1; otherwise, go to Step 4.

Step 4: Let $n = \min\{p, q\}$, and randomly generate a positive integer in [0, n] to initialize $Chr_l^t(\tau)$, $m = m + Chr_l^t(\tau)$. Then, make $t = t + 1$.

Step 5: If $t < T$, return to Step 2; otherwise, output the chromosome $Chr_l(\tau)$, completing the chromosome initialization.

3.2 Genetic operations

(1) Crossover

Based on the basic features of chromosomes, displacement crossover [13, 14] was implemented as follows:

Step 1: Let $P_c(\tau)$ be the crossover probability.

Step 2: Let $l = 1$ (initialize gene t in the chromosome).

Step 3: Generate a random number c in [0, 1]. If $c < P_c(\tau)$, then select $Chr_l(\tau)$ for crossover; otherwise, do not select $Chr_l(\tau)$. Then, make $l = l + 1$.

Step 4: If $l < L$, return to Step 3; otherwise, output the selected chromosome.

Regarding the chromosomes generated by the above process, each infeasible child was repaired by constrained handling [15-17]:

Step 1: Check the chromosome constraints of the child, that is, check the model constraints (3-16).

Step 2: If the child chromosome meets the constraints, then output the child; otherwise, go to Step 3.

Step 3: If the sum of the genes in the child is greater than $\sum_{t=1}^{T} GD_t - FI$, that is, $\sum_{t=1}^{T} chr_l^t(\tau) > \sum_{t=1}^{T} GD_t - FI$, then reduce each non-intersecting gene (i.e. the original gene) of the child by one unit until the sum is no longer greater than $\sum_{t=1}^{T} GD_t - FI$ (*Note:* jump over any non-intersecting gene that equals 0); otherwise, go to Step 4.

Step 4: If the sum of the genes in the child is smaller than $\sum_{t=1}^{T} GD_t - FI$, that is, $\sum_{t=1}^{T} chr_l^t(\tau) < \sum_{t=1}^{T} GD_t - FI$, then increase each non-intersecting gene (i.e. the original gene) of the child by one unit until the sum is no longer smaller than $\sum_{t=1}^{T} GD_t - FI$ (*Note:* jump over any non-intersecting gene that equals TQ_t).

(2) Mutation

Considering the basic features of our model, swapping mutation was performed on the chromosomes:

Step 1: Let $P_m(\tau)$ be the mutation probability.

Step 2: Let $l = 1$ (initialize gene t in the chromosome).

Step 3: Generate a random number r from [0, 1]; if $r < P_m(\tau)$, then select $Chr_l(\tau)$ for mutation; otherwise, do not select $Chr_l(\tau)$. Then, make $l = l + 1$.

Step 4: If $l < L$, return to Step 3; otherwise, output the selected chromosome.

Constrained handling was not applied to mutation, for the chromosomes will still satisfy the constraints, despite the swapping of genes.

3.3 Chromosome evaluation and selection

(1) Chromosome evaluation

The chromosomes were evaluated by the Cheng and Gen's compromise-based individual fitness allocation method [18]:

Step 1: Let $l = 1$.

Step 2: Calculate the values of the four objectives: $LC = \sum_{t=1}^{T}(LQ_t) \times L$, $JC = JQ \times J$, $KC = \sum_{t=1}^{T}(KI_t) \times K$, and $AC = \sum_{t=1}^{T}(AQ_t \times A)$. Then, make $l = l + 1$.

Step 3: If $l < L$, then return to Step 2; otherwise, go to Step 4.

Step 4: Solve the ideal points of the objectives for the τth generation:

$$LC^{\min}(\tau) = \min\{LC_1(Chr_1(\tau)), LC_2(Chr_2(\tau)), ..., LC_l(Chr_l(\tau))\}$$

$$JC^{\min}(\tau) = \min\{JC_1(Chr_1(\tau)), JC_2(Chr_2(\tau)), ..., JC_l(Chr_l(\tau))\}$$

$$KC^{\min}(\tau) = \min\{KC_1(Chr_1(\tau)), KC_2(Chr_2(\tau)), ..., KC_l(Chr_l(\tau))\}$$

$$AC^{\min}(\tau) = \min\{AC_1(Chr_1(\tau)), AC_2(Chr_2(\tau)), ..., AC_l(Chr_l(\tau))\}$$

Step 5: Calculate the regret value of each chromosome of the τth generation:

$$r_l(\tau) = w_1\left|LC_l(Chr_l(\tau)) - LC^{\min}(\tau)\right| + w_2\left|JC_l(Chr_l(\tau)) - JC^{\min}(\tau)\right| + w_3\left|KC_l(Chr_l(\tau)) - KC^{\min}(\tau)\right| + w_4\left|AC_l(Chr_l(\tau)) - AC^{\min}(\tau)\right|,$$

where, $\sum_{i=1}^{4} w_i = 1$.

Step 6: Calculate the maximum and minimum regret points of the chromosomes of the current generation:

$$r_{\max}(\tau) = \max\{r_1(\tau), r_2(\tau), ..., r_L(\tau)\},$$

$$r_{\min}(\tau) = \min\{r_1(\tau), r_2(\tau), ..., r_L(\tau)\};$$

Step 7: Calculate the fitness of each chromosome of the current generation:

$$Fitness(Chr_l(\tau)) = \frac{r_{\max} - r_l(\tau) + \gamma}{r_{\max} - r_{\min} + \gamma}$$

(2) Chromosome selection

The chromosomes of each generation were selected by the elite strategy, using the Boltzmann selector [19]:

Step 1: Calculate the cumulative probability of all chromosomes $q_0 = 0$, $q_i = \sum_{l=1}^{L} p_l(\tau)$, $l = 1, 2, \ldots, L$.

Step 2: Generate a random number r in $(0, q_L]$; if $q_{i-1} < r \le q_i$, then select the lth chromosome $Chr_l(\tau)$.

Step 3: Repeat Step 2 L times to get L chromosomes.

3.4 Automatic parameter adjustment based on fuzzy logic control (FLC)

For optimization problems, one of the key steps of the GA solution [20-24] lies in the optimization of control parameters like crossover probability $P_c(\tau)$ and mutation probability $P_m(\tau)$. During the optimization, different values of control parameters are needed to balance the global and local search abilities of the GA. The specific values depend on the optimization problem. To dynamically adjust the control parameters $P_c(\tau)$ and $P_m(\tau)$, it is necessary to establish an adaptive GA.

In this paper, Yun and Gen's FLC-based adjustment rule is adopted to adjust the control parameters of the GA [25]. Two fuzzy control techniques are employed in this rule: crossover FLC and mutation FLC. During the search for optimal solution, the two fuzzy control techniques can independently and automatically adjust the control parameters, namely, crossover probability $P_c(\tau)$ and mutation probability $P_m(\tau)$.

3.5 Overall framework

To sum up, the complete program for the AT-GA to solve the multi-objective integer programming model is as follows:

Step 1: Input model data, and AT-GA parameters.
Step 2: Let $\tau = 1$.
Step 3: Initialize the chromosomes and determine the initial population $P(\tau)$.
Step 4: Use the regret value method to process the multi-objective model and evaluate the population $P(\tau)$.
Step 5: Perform displacement crossover to generate the child population $C(\tau)$ from the parent population $P(\tau)$.
Step 6: Implement the check and repair procedures to make the child population $C(\tau)$ satisfy model constraints.
Step 7: Perform swapping mutation to generate the child population $C(\tau)$ from the parent population $P(\tau)$.
Step 8: Use the regret value method to process the multi-objective model and evaluate the population $C(\tau)$.
Step 9: Apply Boltzmann selector to choose the chromosomes for the next population $P(\tau+1)$ from the parent population $P(\tau)$ and the child population $C(\tau)$.
Step 10: Implement the automatic transformation strategy to adjust the control parameters $P_c(\tau)$ and $P_m(\tau)$ of the GA.
Step 11: Make $\tau = \tau + 1$.
Step 12: If $\tau < I$, then go back to Step 5; otherwise, output the optimal solution of the model.

4. EXAMPLE ANALYSIS

4.1 Example data

Example data of five different dimensions were collected (Tables I to V), according to the data structure of the proposed MPS optimization model.

Table I: Example data 1.

Items	**Planning horizon *T***			
	1	2	3	4
UQ_t	80	100	150	200
NQ_t	40	20	50	10
TQ_t	120	120	200	210
GD_t	100	150	200	100

Table II: Example data 2.

Items	**Planning horizon *T***					
	1	2	3	4	5	6
UQ_t	150	250	200	150	200	200
NQ_t	50	100	80	70	100	40
TQ_t	200	350	280	220	300	240
GD_t	200	200	200	200	200	200

Table III: Example data 3.

Items	**Planning horizon *T***							
	1	2	3	4	5	6	7	8
UQ_t	240	320	180	370	160	250	100	280
NQ_t	80	100	40	20	100	50	20	0
TQ_t	320	420	220	390	260	300	120	280
GD_t	300	350	200	400	200	200	150	240

Table IV: Example data 4.

Items	Planning horizon T									
	1	2	3	4	5	6	7	8	9	10
UQ_t	180	270	120	250	50	240	200	150	100	190
NQ_t	40	20	50	40	20	80	20	50	20	40
TQ_t	220	290	170	290	70	320	220	200	120	230
GD_t	200	300	150	240	80	280	180	230	120	210

Table V: Example data 5.

Items	Planning horizon T											
	1	2	3	4	5	6	7	8	9	10	11	12
UQ_t	180	180	180	180	180	180	180	180	180	180	180	180
NQ_t	40	40	40	40	40	40	40	40	40	40	40	40
TQ_t	220	220	220	220	220	220	220	220	220	220	220	220
GD_t	200	200	200	200	200	200	200	200	200	200	200	200

4.2 Performance comparison

The proposed AT-GA was compared with three multi-objective GAs, namely, non-dominated sorting genetic algorithm (NSGA), NSGA-II, and adaptive weighted sum genetic algorithm (AWGA). One-dimensional (1D) *t*-test (confidence $\alpha = 0.05$) was carried out to ensure the statistical validity of the comparison. All four algorithms were programmed on Matlab 8.0.0.783 and operated on an Intel Core Duo 2 central processing unit (CPU) (2.53 GHz) and a memory of 4,096 MB. Table VI compares the performance of the four algorithms.

As shown in Table VI, under the confidence of $\alpha = 0.05$, the AT-GA outperformed NSGA, NSGA-II, and AWGA in the quality of non-inferior solutions for multi-objective problems (MP_1), the diversity of non-inferior solutions for multi-objective problems (MP_2), and the calculation time (MP_3). In terms of MP_1, the performance of the AT-GA was at least 3.39 % better than that of the other multi-objective GAs; in particular, the performance advantage over the NSGA was as high as 24.66 %. In terms of MP_2, the AT-GA obtained at least 5.98 % more non-inferior solutions (diversity) than the other algorithms: 37.60 % more than that of NSGA, 18.00 % more than that of NSGA-II, and 5.98 % more than that of AWGA. In terms of MP_3, the AT-GA also boasted a clear advantage, reducing the calculation time by at least 13.49 % from that of the other algorithms. In summary, the AT-GA can effectively handle multi-objective integer programming models.

Table VI: The performance of the four algorithms.

Algorithms	**The percentage of performance advantage of the AT-GA over the other algorithms** (at $\alpha = 0.05$)		
	MP_1	MP_2	MP_3
NSGA	24.66 %	37.60 %	126.39 %
NSGA-II	9.82 %	18.00 %	28.28 %
AWGA	3.39 %	5.98 %	13.49 %

4.3 Adaptability analysis

In the AT-GA, the automatic transformation dynamically adjusts the control parameters via FLC during the operation, which further improves the search ability of the algorithm. To test the degree of improvement, the AT-GA was compared with GA in solving high-dimensional models. The calculation examples are high-dimensional multi-objective MPS optimization models, whose T equals 10, 20, 30, and 40, respectively.

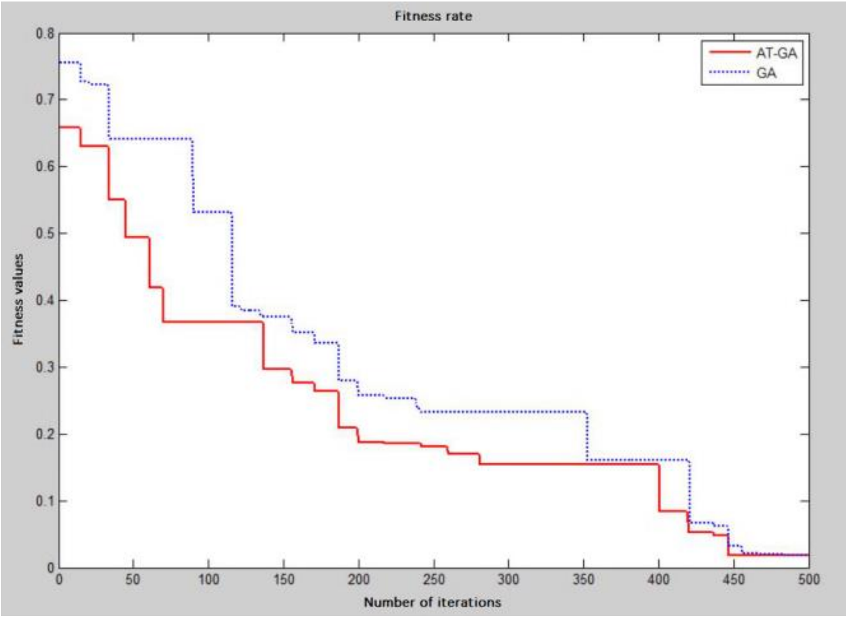

Figure 1: Convergences of AT-GA and GA.

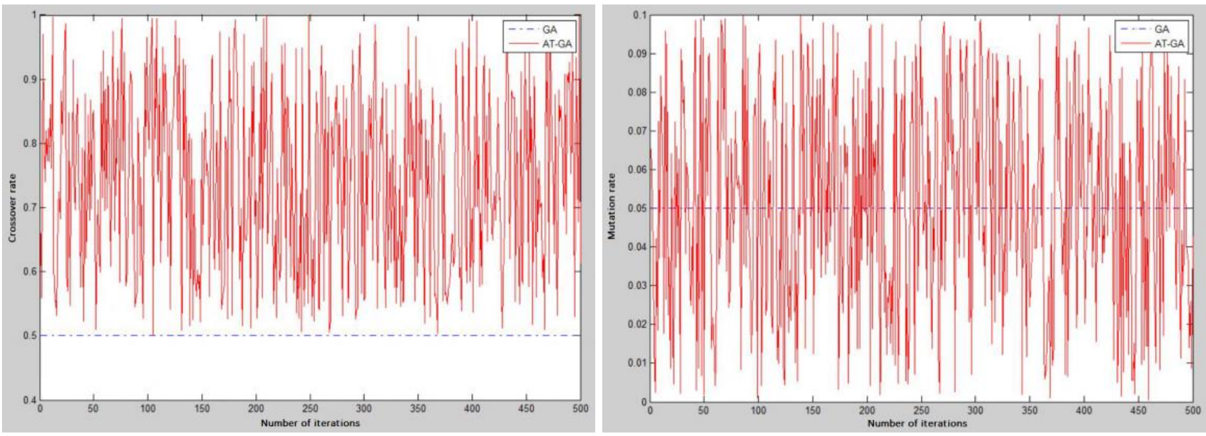

Figure 2: Crossover and mutation performances of AT-GA and GA.

Fig. 1 presents the convergences of the AT-GA and GA at $T = 10$. It can be seen that both the AT-GA and GA converged to the optimal solutions, especially in high-dimensional problems. Fig. 2 compares the crossover and mutation performances of the two algorithms in the search process. The crossover and mutation probabilities of the GA remained the same in the search process, showing no sign of adaptability. By contrast, the two probabilities of the AT-GA changed continuously. This means the AT-GA has more stable adaptability than GA and applies better to real-world multi-objective MPS optimization problems.

5. CONCLUSIONS

Balanced production helps to reduce production waste, balance the load of workers, prevent equipment failures, cut down inventory, lower production cost, and improve product quality. Based on balanced production, this paper proposes a single-product, multi-stage, and multi-objective MLS optimization model. Apart from constraints like net demand and production capacity, the MPS considers a total of four performance management objectives, namely, balanced production, JIT delivery, inventory occupation, and overtime production.

Next, the AT-GA was put forward to solve the multi-objective nonlinear integer programming model: Under the framework of conventional GA, the chromosomes were encoded as integers; an initialization method was designed to ensure the feasibility of initial chromosomes; then, the chromosomes were subject to displacement crossover, and the unfeasible child chromosomes were repaired through constrained handling.

In addition, the Boltzmann selector and FLC were introduced to improve the algorithm. The Boltzmann selector adjusts the search range by intelligently choosing candidate chromosomes. In this way, the early search range is expanded to avoid the local optimum trap; the latter search range is narrowed to improve the efficiency of converging to the optimal solution. The FLC automatically adjusts the crossover and mutation probabilities, making crossover and mutation more effective. The organic integration between the two techniques effectively balances the global and local search abilities of the GA, enhancing the computing power of the algorithm.

Finally, two experiments were carried out to verify the proposed model and algorithm. The experimental results show that the AT-GA is capable of handling multiple objectives and searching the global optimal solution.

REFERENCES

[1] Morita, D.; Suwa, H. (2015). An exact method for robust capacity requirements planning, *International Journal of Automation Technology*, Vol. 9, No. 3, 216-221, doi:10.20965/ijat.2015.p0216

[2] Hung, Y.-F.; Huang, C.-C.; Yeh, Y. (2013). Real-time capacity requirement planning for make-to-order manufacturing with variable time-window orders, *Computers & Industrial Engineering*, Vol. 64, No. 2, 641-652, doi:10.1016/j.cie.2012.11.003

[3] Chen, J. C.; Fan, Y.-C.; Chen, C.-W. (2009). Capacity requirements planning for twin Fabs of wafer fabrication, *International Journal of Production Research*, Vol. 47, No. 16, 4473-4496, doi:10.1080/00207540802007597

[4] Matsuura, H.; Tsubone, H. (1993). Setting planned leadtimes in capacity requirements planning, *Journal of the Operational Research Society*, Vol. 44, No. 8, 809-816, doi:10.1057/jors.1993.139

[5] Pan, E.; Liao, W.; Xi, L. (2010). Single-machine-based production scheduling model integrated preventive maintenance planning, *The International Journal of Advanced Manufacturing Technology*, Vol. 50, No. 1-4, 365-375, doi:10.1007/s00170-009-2514-9

[6] Wang, D.; Zhang, H.; Lu, J. F. (2007). Research and application of APS algorithm based on the bottleneck restriction, *Mechatronics*, Vol. 13, No. 6, 35-39, doi:10.3969/j.issn.1007-080X.2007.06.011

[7] Liu, Z.; Xia, Y. Q.; Zheng, S. F. (2017). Standardized framework of capacity balance for typical production planning of aviation manufacturing industry, *Aeronautic Standardization & Quality*, Vol. 2017, No. 2, 3-7, doi:10.13237/j.cnki.asq.2017.02.001

[8] Li, X.; Qin, T. B. (2019). Solving a balanced vehicle master production plan, *Industrial Engineering and Management*, Vol. 24, No. 5, 127-131, doi:10.19495/j.cnki.1007-5429.2019.05.017

[9] Xu, H.-C.; Xu, X.-F.; Zhan, D.-C.; Wang, H.-Y. (2009). A heuristic procedure of master production scheduling for balanced and optimized utilization of bottleneck capacities, *Journal of Harbin Institute of Technology*, Vol. 41, No. 1, 80-84

[10] Xu, Y.; Xie, Q.; Liu, Q. (2010). The study on balance of production schedule and capability of Deli Molded products Co. LTD, *Proceedings of the 2010 IEEE 17th International Conference on Industrial Engineering and Engineering Management*, 813-816, doi:10.1109/ICIEEM.2010.5646498

[11] Kanet, J. J.; Stößlein, M. (2010). Integrating production planning and control: towards a simple model for capacitated ERP, *Production Planning & Control*, Vol. 21, No. 3, 286-300, doi:10.1080/09537280903363209

[12] Lan, B. X.; Chen, H. N. (2001). An integrated enterprise production planning model, *Systems Engineering Theory and Practice*, Vol. 21, No. 9, 33-41, doi:10.3321/j.issn:1000-6788.2001.09.006

[13] Hosseinabadi, A. A. R.; Vahidi, J.; Saemi, B.; Sangaiah, A. K.; Elhoseny, M. (2019). Extended genetic algorithm for solving open-shop scheduling problem, *Soft Computing*, Vol. 23, No. 13, 5099-5116, doi:10.1007/s00500-018-3177-y

[14] Zhang, R.; Tao, J. (2017). A nonlinear fuzzy neural network modeling approach using an improved genetic algorithm, *IEEE Transactions on Industrial Electronics*, Vol. 65, No. 7, 5882-5892, doi:10.1109/TIE.2017.2777415
[15] Chootinan, P.; Chen, A. (2006). Constraint handling in genetic algorithms using a gradient-based repair method, *Computers & Operations Research*, Vol. 33, No. 8, 2263-2281, doi:10.1016/j.cor.2005.02.002
[16] Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, No. 11-12, 1245-1287, doi:10.1016/S0045-7825(01)00323-1
[17] Gen, M.; Cheng, R. (1996). *Genetic Algorithms and Manufacturing Systems Design*, John Wiley & Sons, New York
[18] Goldberg, D. E.; Samtani, M. P. (1986). Engineering optimization via genetic algorithm, *Proceedings of the 9th Conference on Electronic Computation*, 471-482
[19] Nicklow, J.; Reed, P.; Savic, D.; Dessalegne, T.; Harrell, L.; Chan-Hilton, A.; Karamouz, M.; Minsker, B.; Ostfeld, A.; Singh, A.; Zechman, E. (2010). State of the art for genetic algorithms and beyond in water resources planning and management, *Journal of Water Resources Planning and Management*, Vol. 136, No. 4, 412-432, doi:10.1061/(ASCE)WR.1943-5452.0000053
[20] Gocken, T.; Yaktubay, M. (2019). Comparison of different clustering algorithms via genetic algorithm for VRPTW, *International Journal of Simulation Modelling*, Vol. 18, No. 4, 574-585, doi:10.2507/IJSIMM18(4)485
[21] Jemilda, G.; Baulkani, S. (2018). Moving object detection and tracking using genetic algorithm enabled extreme learning machine, *International Journal of Computers Communications & Control*, Vol. 13, No. 2, 162-174, doi:10.15837/ijccc.2018.2.3064
[22] Yang, M. S.; Ba, L.; Xu, E. B.; Li, Y.; Gao, X. Q.; Liu, Y.; Li, Y. (2019). Batch optimization in integrated scheduling of machining and assembly, *International Journal of Simulation Modelling*, Vol. 18, No. 4, 689-698, doi:10.2507/IJSIMM18(4)CO17
[23] Alaia, E. B.; Harbaoui, I.; Borne, P.; Bouchriha, H. (2018). A comparative study of the PSO and GA for the m-MDPDPTW, *International Journal of Computers Communications & Control*, Vol. 13, No. 1, 8-23, doi:10.15837/ijccc.2018.1.2970
[24] Özdemir, H.; Sever, R.; Polat, Ö. (2019). GA-based optimization of SURF algorithm and realization based on Vivado-HLS, *Traitement du Signal*, Vol. 36, No. 5, 377-382, doi:10.18280/ts.360501
[25] Yun, Y.; Gen, M. (2003). Performance analysis of adaptive genetic algorithms with fuzzy logic and heuristics, *Fuzzy Optimization and Decision Making*, Vol. 2, No. 2, 161-175, doi:10.1023/A:1023499201829