Dynamic Schedule Method Based on Rolling Horizon Procedure for a Discrete Manufacturing Shop with Uncertain Processing Time

Ming Li¹*, Peipei Yang¹, Yiping Yuan¹, Wang Xie¹ and Yi Dai²

¹Academy of Mechanical Engineering, Xinjiang University, Urumuqi, 830047, China
²Xinjiang productivity promotion center, Xinjiang University, Urumqi 830099, China.
*Corresponding author’s e-mail: xj_liming@163.com

Abstract. To solve the rescheduling problem of discrete manufacturing shop with uncertain processing time, an improved ant algorithm based on the rolling horizon procedure is proposed. The dynamic scheduling strategy is based on a hybrid scrolling mechanism driven by events and cycles. Analysing disturbances in the process, we divide them into explicit and implicit disturbances. The length deviation tolerance (LDT) of processing time is designed and proposed to filter out redundant rescheduling. The dynamic scheduling algorithm, based on improved ant colony algorithms, is considered with the resource constraint of the shop, and with the use of a customized state transition rule, it helps to overcome the drawbacks of long ant path searching prone to stagnation. Using simulation, the performance of the dynamic scheduling strategy and scheduling algorithm are analysed and verified, and better rolling scheduling policy parameters are obtained.

1. Introduction
In the actual workshop production process, because of the difference in the worker's skill level, equipment condition and so on, the optimized scheme may not be the best or even no longer feasible when the model established in the deterministic environment is implemented. And there are some deviations in the processing time of each process, the uncertainties of resources will have a direct or indirect impact on schedule. The research on job shop scheduling under uncertain processing time has important theoretical value and practical significance for dealing with the job shop production under uncertain processing time.

Predictive reactive scheduling is a common strategy for discrete manufacturing workshops to reschedule [1]. It includes two basic steps: (1) generating a pre-scheduling scheme without considering the future dynamic events of the workshop layer; (2) updating the pre-scheduling scheme triggered by a certain driving mechanism, and rescheduling can keep the feasibility of the scheduling scheme. Or improve scheduling performance. Rolling scheduling is a kind of scheduling method based on prediction. Fang Jian et al. [2] proposed a rolling scheduling method based on work pieces for Job Shop scheduling problem. The main research contents are how to determine the rolling window and how to re-schedule to ensure the continuity of production. Yamamoto et al. [3] proposed a three-stage rescheduling strategy for real-time control of manufacturing operating system.

Abumizar et al. [4] proposed a rescheduling algorithm. When disturbance occurs, only those jobs that are directly or indirectly affected by disturbance are rescheduled, which reduces the increase of production cycle caused by disturbance and also reduces the deviation from the initial scheduling. Liu
et al. \cite{5} have done a lot of simulation experiments, from the simulation data generated training samples for training neural networks, and the trained neural networks for dynamic scheduling. Jones et al. \cite{6} proposed a hybrid framework for solving real-time scheduling and scheduling problems, which integrates neural network, genetic algorithm and real-time simulation. Chen et al. \cite{7} used a periodic strategy with fixed time interval to schedule, and combined with genetic algorithm, and he used a method based on arbitrary key coding to find a scheduling to minimize the production idle time and delay penalty, and achieved good results. In recent years, swarm intelligence method \cite{8-9} has attracted wide attention because of its universal applicability and low experience complexity. It does not need a lot of probability calculation and has strong parallelism.

Therefore, we take discrete manufacturing shop with uncertain processing time as the object of study. Aiming at the dynamic scheduling problem with uncertain working time, the concept of due-date deviation is proposed and a rolling driving mechanism based on tolerance of due-date deviation is designed to effectively buffer the frequency of dynamic events. In the framework of rolling time domain decomposition, an improved ant colony algorithm is designed to complete the window job rescheduling, so as to construct a systematic and perfect dynamic scheduling method for discrete manufacturing shop under uncertain working time.

2. Description of discrete manufacturing shop scheduling

Random disturbances, affecting the scheduling environment, can be divided into explicit and implicit, according to the influence degree. An explicit disturbance is an event that affects the process, including equipment failure, rush order, and the shortage of scheduling resources. An implicit disturbance is an event that occurs at a time that does not seriously affect the progress of the procedure, but if it occurs often, it seriously disrupts the process, including chaos in processing, the machine's state, and the worker's sudden worsening.

To obtain an effective discrete manufacturing shop scheduling model, we make the following assumptions:

I. There is a pre-estimate of the processing time, and there is a certain deviation between the actual processing time and the estimated processing time.

II. Each procedure can choose any combination of available resources that match its own, and the number of resources required by the procedure is not greater than the number of resources in the selected resource combination.

III. Each resource combination can be used only by one procedure at the same time.

IV. The processing of the procedure cannot be suspended.

V. Rush order inserting has the highest priority.

To describe the problem of discrete manufacturing shop scheduling clearly, we define the following notation:

$I$: the set of procedures to be assigned, \( i \in I \).

$C$: the unit time cost.

$J$: the set of procedure categories, \( j \in J \).

$R$: the set of resource combinations, \( r \in R \), \( r_n \) is the set of the class \( n \)-th resources in the resource combination \( r \), \( N \) is the type of the resource required for procedures, \( n \in N \).

\( \delta_r^{i_n} \): for procedure \( i \), under the resource combination \( r \), the required number of class \( n \)-th resources.

\( K_r \): the number of procedures that can be done with the resource combination \( r \).

\( x_r^{i_k} \): for procedure \( i \), using the resource combination \( r \) and being processed at the penultimate \( k \)-th position.

\( T_{is} \): the processing time of procedure \( i \) in scenario \( s \), \( T_{is} \in [T_i, \overline{T_i}] \), \( S \) is the set of all possible scenarios, \( s \in S \).

\( L_{di} \): the critical value of the LDT of procedure processing.
\( \Omega \): the set of all feasible schedules, \( X \in \Omega \).

Decision variable: If procedure \( i \) belongs to the category \( j \), then \( L_i^j = 1 \), otherwise \( L_i^j = 0 \).

If the resource combination \( r \) can process on the procedure category \( j \), then \( Q_i^j = 1 \), otherwise \( Q_i^j = 0 \).

for \( X \in \Omega \), which is in scenario \( s \in S \), the processing cost is

\[
C(X, s) = \sum_{i \in I} c_k x_k^i T_i^s L_i^j
\]

\( C^* \) represents the cost of optimal scheduling, which is in the \( s \in S \) scenario,

\[
C^* = \min_{X \in \Omega} C(X, s)
\]

Definition 1 The cost of maximum regret for scheduling \( X \) is

\[
Z(X) = \max_{s \in S} \{C(X, s) - C^*_s\}
\]

The goal of scheduling is to find a scheduling scheme that optimizes the worst case scenario, that is, makes the maximum regret value minimal. According to Definition 1, the objective function is

\[
\min_{X \in \Omega} \{Z(X)\} = \min_{X \in \Omega} \max_{s \in S} \{C(X, s) - C^*_s\}
\]

Transforming the model into a mixed integer linear model, we obtain:

\[
\sum_{i \in I} c_k x_k^i T_i^s - C^*_s \leq z, \forall s \in S
\]

According to assumption II, the number of resources required for each procedure cannot be greater than the total number of resources in the resource portfolio meeting the following constraint:

\[
\delta_k \leq r, \forall k \in K, \forall r \in R
\]

Assumption III that, at the same time on the same day, the same resources cannot be used simultaneously in two cases, implies:

\[
\sum_{r \in R} x_{ik} = 1, \forall i \in I
\]

\[
\sum_{r \in R} x_{ik} \leq 1, \forall k \in K, \forall r \in R
\]

To make the dispatch plan realistic, it is necessary to ensure matching between procedures and resources as follows:

\[
x_{ik}^r \leq \sum_j L_i^j Q_j^r, \forall i \in I, \forall k \in K, \forall r \in R
\]

3. Scheduling strategy based on LDT

3.1. Automatically triggered scrolling scheduling strategy

According to the specifics of scheduling, we adopt a hybrid mechanism for scheduling, a cycle driven mechanism during the whole procedure, and only in the presence of an explicit trigger an event driven mechanism is adopted. In this paper, we propose an automatically triggered scrolling scheduling strategy based on the length deviation tolerance (LDT) of procedure processing times. When the tolerance of the actual time and plan time is within a certain range, the scheduling plan remains unchanged, and it realizes the effective separation of the range of influence of scheduling. When the tolerance is out of the range, to respond fast, we present an automatically triggered scrolling scheduling strategy, to cope with frequent length deviations in practical procedure processing.

In the processing of procedure, the occurrence of a disturbance results in a deviation from the actual scheduling and prescheduling. If the scheduling is adjusted whenever such a disturbance occurs, changes
appear frequently, so we establish a buffer mechanism to reduce the impact of minor disturbances, which can prevent the effects of small disturbance events. With different lengths of processing times, the same deviation may have different influence degrees, so the length deviation tolerance (LDT) is proposed using the following expression:

$$d = \left| \frac{PB_{ik} (or PE_{ik}) - B_{ik}}{T_i} \right|$$

(10)

where $B_{ik}$ and $E_{ik}$ represent the predicted start and end time, respectively, with the resource combination $r$ for procedure $i$ at position $k$ of the process; and $PB_{ik}$ and $PE_{ik}$ represent the actual times, respectively. When the calculated value of $d$, the length deviation tolerance, is less than the critical value $L_d$, the scheduling system does not respond to the dynamic perturbation event; otherwise, it adjusts accordingly.

4. Improved ant colony algorithm

4.1. Improved state transition rule

The probability of the $k$-th ant moves from node $r$ to node $s$ is:

$$P_{ij}^{k} = \frac{\left(\tau_{ij}^{k}\right)^{\alpha} \cdot (\eta_{ij})^{\beta} \cdot x_{ij}}{\sum_{j \in \text{allowed}_i} \left(\tau_{ij}^{k}\right)^{\alpha} \cdot (\eta_{ij})^{\beta} \cdot x_{ij}}$$

(11)

$$x_{ij} = \frac{N_{ij}^N}{N_{ij}^N + \frac{\delta m_{ij} \eta_{ij}}{\max(\eta_j)}}$$

(12)

where $N_i$ represents the number of ants; $N$ is the current number of iterations; and $m_{ij}$ is the total number of ants passing through path $(i, j)$. When an iteration goes to a local optimum, the local optimal path pheromone increases. But the increase in the number of ants in the path leads to a decrease in $x$, leading to the growth of pheromone effects on the state transition probability, improving the algorithm’s global search ability. If $m_{ij} \leq N_i N$ and $\eta_{ij}/\max(\eta_j) \leq 1$, then $1 \leq x_{ij} \leq x_{\text{max}} = 1/(1+\delta)$. The parameter $\delta$ can adjust the intensity of $x$. The smaller $\delta$, the larger $x_{\text{min}}$, and the shorter the path. $\tau_{ij}^{k}$ is the pheromone from node $r$ to node $s$; and $\eta_{ij}^{k}$ is the visibility from node $r$ to node $s$. The biased coefficients $\alpha$ and $\beta$ represent the information and visibility coefficients, respectively. They determine the importance of $\tau_{ij}^{k}$ and $\eta_{ij}^{k}$ in the transfer probability. Visibility is calculated by equation (13).

$$\eta_{ij}^{k} = \frac{1}{t_{\text{wait}} + b}$$

(13)

where $t_{\text{wait}}$ represents the waiting time. To avoid the case when visibility $\eta_{ij}^{k}$ tends to infinity, in the denominator $b > 1$ to ensure the effect of visibility on node selection.

4.2. Implementation of scheduling algorithm

The steps of the algorithm are as follows:

Step 1 Define a two-dimensional array: one dimension for ants, and the other dimension for procedures. The elements of the array are dynamic, which record all resources in combinations, including the number of procedures, resource combinations, workpiece waiting times, start and end times.

Step 2 Initialize the ant colony algorithm parameters. Set $\alpha$, $\beta$, $\rho$, the maximum value is $\tau_{\text{max}}$, the minimum value is $\tau_{\text{min}}$, the maximum number of iterations is $I$, the number of ants is $N_0$, and the jump rate is $P_s$. The initial value of pheromone is set to $\tau_{\text{max}}$. 
Step 3 Generate an ant \( a \), select an unplanned procedure \( i \) randomly, and select procedure \( i \) as the first traveller at the ant node. The selected resource combination \( r = \arg \min_{r=1,2,\ldots,R} \sum_{j} q_{jk}^i T_j^i \) satisfies the constraints \( x_{ik}^r \leq \sum_j q_{jk}^i r_{xj} \). Initialize the ant node counter \( S=1 \), and the loop counter \( N=1 \).

Step 4 Resources \( R_a^r \) that can be travelled by ant \( a \) are ascertained, where \( R_a^r = \{1,2,\ldots,I\} - r_{aba}(s) \), and \( r_{aba}(s) \) is the collection of resource combinations that were visited by ant \( a \) in step \( s \), and \( I \) is the total number of procedures waiting to be assigned. Calculate the weighted divergence of the ant's node, and determine the optional path set \( R_a^r \) for the ant \( a \).

Step 5 According to the ant state transition rule, select a node to travel by the ant in the next step, where the resource combination of the earliest operation can be used as the corresponding resource combination for the operation of the node; update the ant node counter \( S=S+1 \), and the loop counter \( N=N+1 \).

Step 6 Check whether the ant has travelled all nodes, that is, whether \( S=p \), and if it is, then go to the next step; otherwise go to Step 4.

Step 7 Update the number of algorithm cycles \( N=N+1 \). Determine whether \( N=I \), and if it is, then stop; otherwise go to Step 8.

Step 8 Update pheromone concentrations. First, the pheromone is volatilized using formula
\[
\tau_{i,j}(N+1) = \rho \tau_{i,j}(N),
\]
and then, it is added to the path that has the shortest completion time in the cycle using formula
\[
\tau_{i,j}^{a(min)} = \tau_{i,j}^{a(min)} + \Delta \tau_{i,j}^{best},
\]
where \( a(min) \) is the shortest path, \( L_{max}^{best} \) is the search path length of \( a(min) \), and \( \Delta \tau_{i,j}^{best} = 1 / L_{max}^{best} \). To avoid premature convergence to a local optimum, the ant system limits each possible path pheromone concentration to \([\tau_{min}, \tau_{max}]\).

5. Simulation analysis

5.1. Parameter settings
For each procedure, \( T_i(i=1,2,\ldots,I) \) is taken in \([60, C]\) according to the uniform distribution, \( \bar{T}_i(i=1,2,\ldots,I) \) in \([T_i, T_i+C]\) uniformly, \( I \in \{10, 30, 50\} \), \( C \in \{150, 250, 350\} \), \( J \in \{2, 6\} \), and \( R \times J \leq I/2 \). The combinations of the above parameters \( I, C, J, R \) produce 42 cases, where 10 instances were generated randomly in each case, so in total 420 experiments are used. Each value obtained for each case is the average of the corresponding 10 instances.

5.2. Dynamic scheduling parameter test
To obtain the most suitable LDT, we use the 4-stage Erlang distribution of the actual surgical time in simulation, where each case is simulated 10 times with different tolerance using \( \{I, C, J, R\} = \{30, 350, 2, 4\} \). The maximum completion time, rolling times, objective function values and resource utilization are recorded. It can be seen from the table 1, when the LDT is greater than 0.25, there is no rescheduling, and the calculation results are not good. When the LDT is too small, there is a significant increase in the number of rolling schedules, resulting in frequent rescheduling. When the LDT is around 0.1, the number of rescheduling is low, the resource utilization rate is highest, and the maximum value of the objective function is smallest.

| LDT  | Rescheduling times | Completion time (min) | Objective function value \( Z(X) \) | Resource utilization (%) |
|------|---------------------|-----------------------|-------------------------------------|--------------------------|
| 0.025| 20                  | 466.2                 | 621.5                               | 72.1                     |
| 0.050| 11                  | 450.5                 | 580.2                               | 73.6                     |
| 0.075| 6                   | 425.7                 | 556.7                               | 75.7                     |
0.100 4 411.5 535.9 80.3
0.125 3 417.6 546.5 78.5
0.150 3 417.6 546.5 78.5
0.200 2 432.6 557.2 75.6
0.250 1 442.5 620.5 72.3
0.300 1 442.5 620.5 72.3
0.350 0 459.5 710.3 68.2
0.400 0 459.5 710.3 68.2
0.450 0 459.5 710.3 68.2
0.500 0 459.5 710.3 68.2

For the cycle driven mechanism, the number of rolling schedules and the ability of the system to adapt to dynamic factors are determined by the rolling time window $\Delta T$. We fix the size of the experiment, set different steps to test, verify $\Delta T$ and the optimization of the scale of the correlation. From the results of Table 2, we can see that when $\Delta T = 90 \sim 100$ and the number of rescheduling times is 4 ~ 5, the overall effect of dynamic scheduling is close to the best one, the maximum value of the objective function is lowest, and the resource utilization rate is highest.

Table 2. Comparison of rolling time window size scheduling results

| $\Delta T$ | Rescheduling times | Completion time (min) | Objective function value $Z(X)$ | Resource utilization (%) |
|-----------|---------------------|-----------------------|---------------------------------|--------------------------|
| 10        | 48                  | 669.2                 | 754.6                           | 65.1                     |
| 20        | 24                  | 550.5                 | 672.3                           | 65.4                     |
| 30        | 16                  | 481.7                 | 611.4                           | 67.9                     |
| 40        | 12                  | 472.5                 | 596.7                           | 68.4                     |
| 50        | 9                   | 460.6                 | 580.5                           | 70.5                     |
| 60        | 8                   | 455.3                 | 572.4                           | 71.3                     |
| 70        | 7                   | 450.6                 | 563.5                           | 72.4                     |
| 80        | 6                   | 436.5                 | 548.1                           | 73.7                     |
| 90        | 5                   | 411.9                 | 535.6                           | 82.6                     |
| 100       | 4                   | 417.7                 | 536.8                           | 78.7                     |
| 200       | 3                   | 459.5                 | 546.3                           | 69.5                     |
| 300       | 2                   | 511.5                 | 557.2                           | 68.5                     |
| 400       | 1                   | 625.7                 | 620.5                           | 59.5                     |

5.3. Feasibility analysis of dynamic scheduling strategy

A large number of simulation experiments are carried out to obtain the maximum regret value of the scheduling targets with different rescheduling times in the dispatching cycle. The correlation between the rescheduling time and the maximum regret value of the scheduling objective is determined using regression analysis, and then the influence degree on the dynamic scheduling target is studied by means of quantitative analysis. Since the curve type is unknown before data analysis, a polynomial regression is used:

$$y_i = \alpha + \beta_1 x_1 + \beta_2 x_1^2 + \cdots + \beta_k x_1^k + \varepsilon_i$$

where $x_1$ indicates the time of rescheduling, $y_i$ represents the maximum regret value at the time of the current rescheduling, and $\beta_1>0, \beta_2>0, \ldots, \beta_k>0$. Let $x_1=x_1, x_2=x_1^2, \ldots, x_k=x_1^k$, then Equation (14) becomes a linear equation:

$$y_i = \alpha + \beta_1 x_1 + \beta_2 x_1^2 + \cdots + \beta_k x_1^k + \varepsilon_i$$

Using the least square method, the regression curve whose sum of squares of residues $\varepsilon_i \sim N(0, \sigma^2)$ is lowest, is found by minimizing $s_i = \sum_{i=1}^{n} (y_i - a - b_1 x_1 - b_2 x_1^2 - \cdots - b_k x_1^k)^2$, to make $s_i$ minimal, so that
\[
\frac{\partial s_{s_i}}{\partial b_1} = 0 \quad \frac{\partial s_{s_i}}{\partial b_2} = 0 \quad \frac{\partial s_{s_i}}{\partial b_k} = 0 
\] 

(16)

After solving the system, we obtain

\[
b_j = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} 
\]

(17)

\[
a = \frac{1}{n} \left( \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} b_j \sum_{i=1}^{n} x_i \right) 
\]

(18)

After 50 simulation tests, the regression curve was obtained, as shown in figure 1. According to the least square method, it can be shown very easily form Equs. (17) and (18) that \( \alpha = 715.3, \beta_1 = -1.535, \beta_2 = 0.003026 \), and the regression equation is \( y = 715.3 - 1.535x + 0.003026x^2 \).

![Figure 1 Maximum regret value regression curve](image)

According to figure 1, the regression curve of the rescheduling time vs the maximum regret value is a quadratic polynomial. If the initial deviation occurs in an early period, general scheduling is maintained, because the LDT is generally buffered during the free time of the preceding activities and succeeding activities. Rescheduling is not necessary during the early period, it only increases the cost. If the LDT occurs in a late period, this also tends to maintain the original scheduling program, because most of basic tasks at this time are to complete the original scheduling program, so losses in the process of adjustment are difficult to compensate by the benefits of the new program.

5.4. Performance test of dynamic scheduling algorithm

Individual figures should normally be centred but place two figures side-by-side if they will fit comfortably like this as it saves space. Place the figure as close as possible after the point where it is first referenced in the text. If there are a large number of figures it might be necessary to place some before their text citation. Figures should never appear within or after the reference list.

6. Conclusion

(1) A dynamic scheduling strategy, based on a hybrid scrolling mechanism driven by events and cycles, is proposed. After analyzing disturbances in the operation process, we divide them into the explicit and implicit ones. During surgery with the real-time detection of disturbances, it is generally difficult to detect implicit disturbances, so we consider a cycle driven mechanism during the whole process, and only when explicit disturbances occur, an event driven rolling scheduling mechanism is triggered. We design and propose the length deviation tolerance (LDT) of processing times, to filter out unnecessary rescheduling.

(2) A dynamic scheduling algorithm, based on an improved ant colony algorithm, is proposed. Considering resource constraints of operation characteristics, we amend the state transition rule of the
traditional ant colony algorithm, overcoming the shortcomings of the algorithm, namely, long time search and being prone to stagnation.

(3) Using simulations, the parameters of the rolling scheduling strategy are analyzed, and the performance of the dynamic scheduling algorithm is verified. The optimal rolling scheduling parameters are obtained, and the algorithm is proved to be better in the terms of calculation time and calculation results.

Acknowledgments
This work is supported by the National Natural Science Foundation of P.R. China (ID: 71961029).

References
[1] YUAN-GENGHUANG, Kanal, L. V., & Tripathi, S. (1990). Reactive scheduling for a single machine: problem definition, analysis, and heuristic solution. International Journal of Computer Integrated Manufacturing, 3(1), 6-12.
[2] Jian, F., & Yugeng, X. (1997). The genetic algorithms-based rolling horizon scheduling strategy. Control Theory & Applications.
[3] Yamamoto, M., & Nof, S. Y. (1985). Scheduling/rescheduling in the manufacturing operating system environment. International Journal of Production Research, 23(4), 705-722.
[4] Abumaizzer, R. J., & Svestka, J. A. (1997). Rescheduling job shops under random disruptions. International Journal of Production Research, 35(7), 2065-2082.
[5] Liu, H., & Dong, J. (1996). Dispatching rule selection using artificial neural networks for dynamic planning and scheduling. Journal of Intelligent Manufacturing, 7(3), 243-250.
[6] ALBERTJONES, LUISRABELO, & YUEHWERNYIH. (2007). A hybrid approach for real-time sequencing and scheduling. International Journal of Computer Integrated Manufacturing, 8(2), 145-154.
[7] Chen, K. J., & Ji, P. (2007). A genetic algorithm for dynamic advanced planning and scheduling (daps) with a frozen interval. Expert Systems with Applications, 33(4), 1004-1010.
[8] Ju Q Y, Zhu J Y. (2007). Multi-objective optimization of batch production soft job shop scheduling[J]. Chinese Journal of Mechanical Engineering, 43(8):148-154.
[9] Charalambous, C., Fleszar, K., & Hindi, K. S. (2010). A Hybrid Searching Method for the Unrelated Parallel Machine Scheduling Problem. Artificial Intelligence Applications and Innovations - Ifip Wg 12.5 International Conference, Aiai 2010, Larnaca, Cyprus, October 6-7, 2010. Proceedings (Vol.339, pp.230-237). DBLP.
[10] Wang L, Zheng H Y, (2014). Zheng X. Survey on resource-constrained project scheduling under uncertainty[J]. Control and Decision, 29(4):57-514.
[11] Koulinas, G., Kotsikas, L., and Anagnostopoulos, K. (2014). A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem. Information Sciences, 277, 680-693.
[12] Afzalirad M, Shafipour M. (2018). Design of an efficient genetic algorithm for resource-constrained unrelated parallel machine scheduling problem with machine eligibility restrictions. Journal of Intelligent Manufacturing, 1,1-15.
[13] Silvente J, Kopanos G M, Pistikopoulos E N, et al. (2015). A rolling horizon optimization framework for the simultaneous energy supply and demand planning in microgrids[J]. Applied Energy, 155:485-501.
[14] Bradford E, Schweidtmann A M, Lapkin A. (2018). Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm[J]. Journal of Global Optimization, 71(2):1-33.
[15] Kostesha N, Willquist K, Emmeus J, et al. (1997). Robust scheduling with processing time uncertainty[J]. Computers & Chemical Engineering, 21(10): S1055–S1060.
[16] QidongCao, Patterson J W, Griffin T E. (2001). On the operational definition of processing time uncertainty[J]. International Journal of Production Research, 39(13):2833-2849.