ISSN: 2688-8645

Multi-Objective Optimization Method for Dynamic Scheduling in 3D Printing Based on Improved NSGA-II

Longkun Luo [1,*], Liang Guo [1,2], and Lixu Mou [1]

[1] School of Mechatronic Engineering, Southwest Petroleum University, Chengdu, Sichuan, 610500, China

[2] Sichuan Shared Platform for Oil and Gas Equipment Technology, Chengdu, Sichuan, 610500, China

* Corresponding author: Longkun Luo

Abstract

To address unreasonable scheduling in distributed 3D printing systems under dynamic disturbances, an event-driven multi-objective dynamic scheduling optimization method is proposed. First, a model minimizing makespan and total processing cost is constructed, defining disturbance triggers and rescheduling criteria. Second, an improved NSGA-II algorithm is designed, featuring a chromosome segmentation mechanism to coordinate local and global optimization, adaptive operators for convergence efficiency, and a cost-aware greedy mutation strategy to enhance solution quality. Simulations show that, compared to traditional right-shift rescheduling, the proposed method significantly reduces makespan and slightly lowers total cost in equipment failure scenarios, offering an efficient solution for cloud manufacturing platforms.

Keywords

3D Printing; Dynamic Scheduling; Multi-Objective Optimization; NSGA-II.

1. Introduction

With the rapid development and wide application of additive manufacturing technology [1], 3D printing has gradually expanded from prototype manufacturing to end-product production, personalized customization services, and distributed manufacturing systems. In this process, the scheduling optimization problem [2] of 3D printing systems has become increasingly prominent, especially when facing dynamic disturbance environments, where traditional static scheduling methods are difficult to adapt to actual production needs.

Aiming at the multi-objective job shop scheduling problem in complex manufacturing environments, Zeng et al. [3] proposed the NSGA-III-SD algorithm to determine the optimal execution sequence of a set of jobs on multiple machines to maximize production efficiency; Ouyang Hongcai et al. [4] aimed at the low efficiency of solving multi-objective flexible job shop scheduling, proposed an improved NSGA-III scheduling optimization algorithm, and designed a hybrid allocation strategy; Li Xiaohui et al. [5] proposed a meta-heuristic algorithm combining an improved genetic algorithm with neighborhood search technology, minimizing the maximum completion time of all tasks as the optimization goal, effectively solving the dynamic rescheduling problem in cloud manufacturing environments. Aiming at the task scheduling challenges faced by 3D printing in distributed cloud manufacturing, industrial internet, and multi-robot collaborative environments, Darwish et al. [6] faced personalized 3D printing tasks in IIoT environments, proposed a green real-time multi-task scheduling architecture, designed an adaptive priority scheduling algorithm, and showed superior performance and scalability under high loads; He et al. [7] aimed at the multi-dynamic disturbance problems faced by 3D printing cloud platforms in green scenarios, constructed a scheduling model considering unit

ISSN: 2688-8645

mass cost and carbon emissions, and proposed a heuristic strategy; Poudel et al. [8] aimed at collision avoidance and resource constraint problems in multi-robot collaborative 3D printing, proposed a dynamic dependency list algorithm and an improved genetic algorithm, with the latter having advantages in heterogeneous tasks; He et al. [9] faced 3D printing cloud platforms in medical scenarios, proposed a real-time scheduling strategy integrating dynamic mechanisms and an improved multi-objective grey wolf algorithm, effectively reducing comprehensive costs and delivery time.

This paper focuses on typical dynamic disturbances in the 3D printing process: equipment failure, and studies event-driven dynamic scheduling strategies. By constructing a multi-objective optimization model that balances makespan and total processing cost, an improved NSGA-II algorithm is proposed, introducing chromosome segmentation mechanisms, adaptive genetic operators, and cost-based greedy mutation strategies to achieve rapid response and global optimization of disturbance events.

2. Problem Modeling

The 3D printing dynamic scheduling problem can be described as: multiple 3D printing tasks are executed on multiple 3D printers with different characteristics in different locations. Each printing task is processed as a complete unit, and at least one printing task can be executed on multiple compatible 3D printers. Tasks may arrive dynamically at any time, and production needs to be arranged in real-time under certain optimization objectives.

2.1. Dynamic Scheduling Strategy Selection

In dynamic scheduling, attention needs to be paid to the trigger mechanism of rescheduling and the rescheduling method adopted [10] . Common ways to trigger rescheduling include periodic rescheduling [11] , event-driven rescheduling [12] , and hybrid rescheduling. Periodic rescheduling recalculates and updates the scheduling scheme regularly according to fixed time intervals. Event-driven rescheduling triggers rescheduling only when specific events occur, such as machine failure, task arrival, resource failure, etc. Hybrid rescheduling combines periodic and event-driven strategies, checking at fixed cycles while also responding immediately to key events. In the actual 3D printing process, disturbance events usually do not occur frequently within a short period. Therefore, event-driven rescheduling is usually selected in dynamic scheduling, that is, rescheduling is performed when a dynamic event occurs.

According to the adjustment range and strategy, taking equipment failure disturbance events as the main research factor, right-shift rescheduling and full rescheduling are adopted respectively for effect comparison. Right-shift rescheduling is the simplest rescheduling strategy. After a disturbance occurs, the start times of the affected tasks and their subsequent tasks are shifted backward as a whole, without changing the original execution sequence and resource allocation of tasks. This method has extremely low computational overhead, is simple to implement, and can quickly restore scheduling feasibility. Full rescheduling treats all unexecuted tasks after the current moment as a new scheduling problem every time a disturbance occurs, and re-optimizes the entire scheduling scheme from the beginning. This method can maximize the use of the latest information to generate high-quality scheduling, having optimal adaptability and performance potential.

2.2. Dynamic Scheduling Model Construction

Aiming at the distributed 3D printing service scheduling problem. In this scenario, manufacturing resources are distributed as service nodes in different geographical locations. Each 3D printing task can be selected from multiple heterogeneous 3D printers. The scheduling problem needs to consider two conflicting objectives simultaneously: makespan and total processing cost, with the following conditions:

ISSN: 2688-8645

(1)The same 3D printer can only execute one printing task at the same time;
(2)At the same time, a printing task can only be executed on one 3D printer;
(3)There is no priority difference between different printing tasks by default, but there may be different delivery date requirements, and different priorities can be set according to urgency;
(4)Each 3D printer has specific material compatibility, build size, and precision limits, and can only accept printing tasks that meet its characteristics;
(5)The printing tasks in the system at the initial moment can all be scheduled, and newly arrived tasks need to dynamically adjust the scheduling scheme according to the current system state.

2.2.1. Symbol Definition

Table 1. Definition of symbols

Symbol	Description	Remarks
J	Set of 3D printing parts, $J=\{J_0,J_1,\ldots,J_{n-1}\}$	Total of n printing tasks
P	Set of 3D printers, $P=\{P_1,P_2,\ldots,P_m\}$	Total of m printers
F	Set of factories, $F=\{F_1,F_2,\ldots\}$	Printers belong to different factories
$T_{i,k}^{P}$	Printing duration of part $J_i$on printer P_k	Printing processing duration
Tr_k	Logistics transportation duration of the factory to which printer P_k belongs	Logistics transportation duration
W_k	Total cumulative working hours of printer P_k	$W_k=\sum_i T_{i,k}^{P}\cdot x_{i,k}$
$x_{i,k}$	If task J_i is assigned to printer P_k for processing, it is 1, otherwise 0	0-1 Variable
$y_{i,j,k}$	If tasks J_i and J_j are both assigned to printer P_k, and J_i is processed before J_j, it is 1, otherwise 0	0-1 Variable
T_i^{S}	Start processing time of task J_i	Continuous Variable
T_i^{E}	Processing completion time of task J_i	Continuous Variable
T_i^{C}	Final delivery time of task J_i (including logistics time)	$T_i^{C}=T_i^{E}+Tr_k$
T_{repair}	Machine repair duration	
T_{break}	Occurrence time of failure or insert order	
C_k^{unit}	Unit time processing cost of machine P_k	
C_k^{trans}	Logistics transportation cost of the factory to which machine P_k belongs	

2.2.2. Mathematical Model

Objective Functions:
Minimize the final delivery time of all tasks:

$$\min f_1=\max_{i\in J}\left\{T_i^{S}+\sum_{k\in P}x_{i,k}\cdot\left(T_{i,k}^{P}+Tr_k\right)\right\} \tag{1}$$

Where $\sum_{k\in P}x_{i,k}\cdot T_{i,k}^{P}$ is the actual processing time, and $\sum_{k\in P}x_{i,k}\cdot Tr_k$ is the logistics transportation time of the corresponding factory.
Minimize total cost:

$$\min f_2=\sum_{k\in P}\left(W_k\cdot C_k^{unit}\right)+\sum_{i\in J}\sum_{k\in P}\left(x_{i,k}\cdot C_k^{trans}\right) \tag{2}$$

Where the first term is the processing cost, and the second term is the logistics cost.
Constraints:

ISSN: 2688-8645

1. Printer Allocation Constraint:

Each task must be assigned to one and only one 3D printer with processing capability:

$$\sum_{k \in M_i} x_{i,k} = 1, \quad \forall i \in J \tag{3}$$

2. Processing Time and Logistics Time Constraint:

The processing completion time and final delivery time of tasks are defined as follows:

$$T_i^E = T_i^S + \sum_{k \in M_i} T_{i,k}^P \cdot x_{i,k}, \quad \forall i \in J \tag{4}$$

$$T_i^C = T_i^E + \sum_{k \in M_i} Tr_k \cdot x_{i,k}, \quad \forall i \in J \tag{5}$$

3. Sequence Constraint:

Tasks on the same machine cannot be executed overlappingly. For any two different tasks J_i and J_j on machine P_k:

$$T_j^S \geq T_i^E - L \cdot (3 - x_{i,k} - x_{j,k} - y_{i,j,k}) \tag{6}$$

$$T_i^S \geq T_j^E - L \cdot (2 - x_{i,k} - x_{j,k} + y_{i,j,k}) \tag{7}$$

Where L is a sufficiently large positive number. This constraint ensures that if i, j are both processed on k and I is before j, then the start time of j must be later than the processing end time of i.

4. Variable Non-negative Constraint:

$$T_i^S \geq 0, \quad \forall i \in J \tag{8}$$

5. Processed Parts Constraint:

For tasks that have been completed or are in progress, keep their original allocation and start time unchanged:

$$x_{i,k} = x_{i,k}^{old}, \quad T_i^S = T_i^{S,old}, \quad \forall i \in H \tag{9}$$

6. Unprocessed Parts Constraint:

For affected future tasks, their start time must not be earlier than the rescheduling trigger moment (or plus failure repair time):

$$T_i^S \geq T_{break}, \quad \forall i \in U \tag{10}$$

If machine P_k fails, the repair time is T_{repair}, and $x_{i,k} = 1$, then:

$$T_i^S \geq T_{break} + T_{repair} \tag{11}$$

3. Improved NSGA-II Algorithm Design

The problem studied in this chapter needs to consider two optimization objectives simultaneously, so a multi-objective optimization algorithm is needed to solve this problem. Non-dominated Sorting Genetic Algorithm II (NSGA-II) has excellent performance and theoretical foundation and has become a milestone achievement in the field of multi-objective optimization.

3.1. Chromosome Encoding and Decoding

In order to determine machine allocation and the processing sequence on each machine, the algorithm adopts a dual-layer encoding scheme based on Operation Sequence (OS) and Machine Sequence (MS).

Operation Sequence (OS): Uses integer encoding. In the 3D printing scenario, OS is the priority queue of printing tasks. Whoever is ranked first is considered for allocation first.

ISSN: 2688-8645

Machine Allocation Sequence (MS): The length of MS is the same as OS. The *k*-th bit of MS represents the machine selected by the *k*-th workpiece in OS. Since the optional machine set for each workpiece is different, MS stores the index in the optional machine set or directly stores the machine ID.

3.2. Improved Algorithm Design

3.2.1. Dynamic Rescheduling Mechanism

Aiming at dynamic events of machine failure, an event-driven rescheduling strategy is adopted. In order to ensure the stability of the production site, the status of workpieces that have been completed or are in progress must be preserved.

Chromosome Segmentation Strategy:

At the rescheduling moment T_{res}, the algorithm first divides all workpieces into two categories according to the current system state:

(1)History Set: Workpieces with start time $S_{ij} < T_{res}$. This part of the process is regarded as "frozen", and its machine allocation and processing sequence remain unchanged during the rescheduling process.

(2)Future Set: Workpieces with start time $S_{ij} \geq T_{res}$. This part of the workpieces is the optimization object of rescheduling.

In the algorithm implementation, by traversing the Gantt chart data of the current optimal solution, the split point split_index is determined. In subsequent crossover and mutation operations, only the gene fragment after split_index is disturbed, thereby strictly constraining the scope of rescheduling.

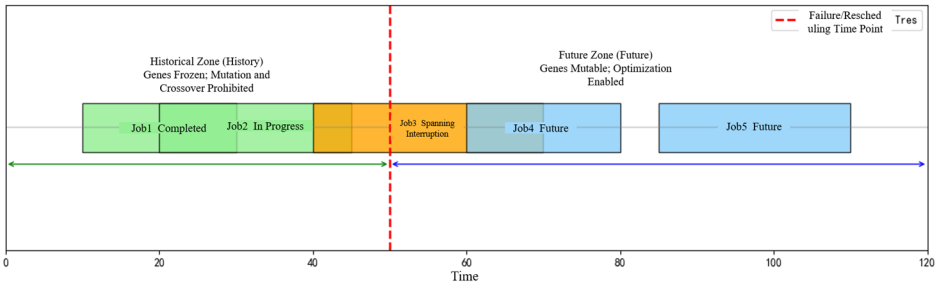

Fig 1. Chromosome segmentation mechanism

3.2.2. Improved Genetic Operators

Since the standard NSGA-II has problems such as slow convergence speed in the later stage and easy to fall into local optima, the following improvement strategies are introduced:

1. Adaptive Crossover and Mutation Probability

The algorithm dynamically adjusts the crossover probability P_c and mutation probability P_m according to the evolution process of the population. The crossover probability decreases linearly with the increase of generations.

$$P_c(t) = P_{c_{max}} \times (1 - \alpha \times \frac{t}{T_{max}}) \tag{12}$$

$\alpha = 0.4$ .In the early stage of evolution, a higher crossover probability helps the population to conduct extensive global search in the solution space; in the later stage of evolution, a lower crossover probability helps to retain the already formed excellent sub-structures and prevent destroying the pattern of the global optimal solution.

The mutation probability increases linearly with the increase of generations.

ISSN: 2688-8645

$$P_m(t) = P_{m_{min}} + \beta \times \frac{t}{T_{max}} \quad (13)$$

$\beta = 0.2$. Since a greedy mutation strategy is adopted, the mutation operation has strong local search properties. Increasing the mutation probability in the later stage of evolution actually enhances the algorithm's refined search ability for local solutions, helping to jump out of local optima and approach the Pareto front.

2. Cost-based Greedy Mutation

Standard algorithms usually adopt random mutation or greedy mutation based on shortest processing time. In order to strengthen the optimization of the total cost objective, the improved algorithm introduces a cost-based greedy strategy at the machine allocation layer. The specific steps are as follows:

(1)Select a gene bit (workpiece).

(2)Obtain all optional machine sets $\{M_1, M_2, ..., M_k\}$ for the workpiece.

(3)Calculate the processing cost on each machine: $Cost_r = Time_r \times UnitCost_r$.

(4)Select the machine with the minimum $Cost_r$ to replace the original machine.

This strategy directly integrates cost information into the evolution process, allowing the population to converge quickly to the low-cost region. In the 3D printing scenario, this will make the algorithm balance printing speed and machine usage cost.

3. POX Crossover

In the crossover operation of the operation sequence (OS), the POX crossover operator is used to maintain the priority constraints of the process. The steps are as follows:

(1)Randomly divide the workpiece set into two subsets S_1 and S_2.

(2)Offspring C_1 inherits the workpiece genes belonging to S_1 from parent P_1, and the positions remain unchanged.

(3)The remaining positions of offspring C_1 are filled with workpiece genes belonging to S_2 from parent P_2 in order.

(4)Perform symmetric operations on C_2.

In the rescheduling scenario, POX crossover is only applied to the Future part of the chromosome, ensuring that the solution generated by rescheduling still satisfies the process constraints.

4. Simulation Experiments

4.1. Experimental Design

Table 2. Machine parameters

Factory Name	Machine Range (P_ID)	Logistics Time (Tr)	Logistics Cost (C^{trans})
Factory A	P1-P5	2	10
Factory B	P6-P10	4	20
Factory C	P11-P15	5	25
Factory D	P16-P20	3	15

Table 3. Machine failure simulation parameters

Machine Failure Scenario		
Fault Machine P_{fault}	Set No. 12 machine failure	error_M = 12
Failure Time t_{break}	Set at time 10	error_S = 10
Repair Time T_{repair}	Set as 5 time units	error_T = 5

ISSN: 2688-8645

Table 4. Parameters of the improved NSGA-II algorithm

Parameter	Improved NSGA-II
Population Size	100
Iterations	50
Crossover Probability	0.8
Mutation Probability	0.2

Table 5. Experimental dataset

Job ID	Optional Machines Count	Machine ID	Proc. Time	Machine ID	Proc. Time	Machine ID	Proc. Time	Machine ID	Proc. Time
1	2	3	2	8	8				
2	4	20	1	10	3	2	6	5	4
3	2	6	2	18	1				
4	2	15	9	19	2				
5	4	20	5	9	6	12	1	19	4
6	2	18	8	8	4				
7	4	13	9	17	10	1	9	2	5
8	3	20	2	17	10	16	8		
9	4	16	5	9	2	8	6	12	8
10	3	15	6	16	10	10	4		
11	4	1	4	10	10	3	8	4	2
12	3	19	7	10	6	12	4		
13	3	16	8	4	10	8	2		
14	3	6	7	10	1	20	10		
15	4	9	1	20	7	18	9	19	6
16	4	8	5	15	5	3	8	14	3
17	2	9	6	13	7				
18	2	4	1	8	4				
19	3	12	7	8	4	4	4		
20	3	5	7	10	3	7	5		
21	2	2	8	8	1				
22	3	16	3	3	8	19	10		
23	4	6	9	16	8	5	10	3	6
24	3	12	3	6	3	7	10		
25	2	15	6	19	1				
26	3	19	4	2	5	12	4		
27	3	12	6	9	8	10	6		
28	2	3	3	10	1				
29	4	6	6	2	4	3	6	13	2
30	4	4	9	8	6	20	9	10	2
31	3	5	1	16	6	19	9		
32	4	13	7	2	9	15	3	3	6
33	3	15	5	8	6	20	2		
34	3	16	7	11	2	13	10		
35	2	14	1	6	1				
36	2	12	7	1	2				
37	4	16	10	13	8	18	6	10	7
38	4	8	4	4	4	16	8	9	1
39	3	5	9	15	10	13	10		
40	3	10	9	2	5	4	2		
41	3	16	2	8	8	3	4		
42	3	19	10	10	7	7	9		
43	3	9	2	18	10	7	9		
44	2	4	10	12	4				
45	4	14	10	20	6	1	1	5	4
46	3	15	7	4	3	14	8		
47	4	19	3	12	6	2	7	8	5
48	3	19	5	12	7	4	10		
49	2	1	8	12	10				
50	2	1	7	19	2				

ISSN: 2688-8645

In order to simulate a multi-factory collaborative manufacturing environment, the experiment allocates 20 3D printers to 4 different factories, each factory having different logistics transportation times.

4.2. Result Analysis

Simulations involved 20 printers and 50 jobs, with an initial makespan of 51. A failure occurred on Printer 12 at time 10, lasting 5 units. Traditional right-shift rescheduling passively delayed affected tasks, increasing makespan to 56 and causing resource idleness. The proposed full rescheduling strategy reallocated affected jobs to available printers and collaboratively optimized logistics and processing sequences, significantly compressing makespan to 21, greatly shortening the cycle and improving utilization. Convergence analysis shows makespan stabilized at 21 around generation 18, while the cost curve showed step-like declines, stabilizing below 3000 after generation 40. Unlike right-shift which retains original costs but delays delivery, the proposed method uses a cost-aware greedy mutation strategy to select lower-cost machines, achieving dual optimization of timeliness and economy.

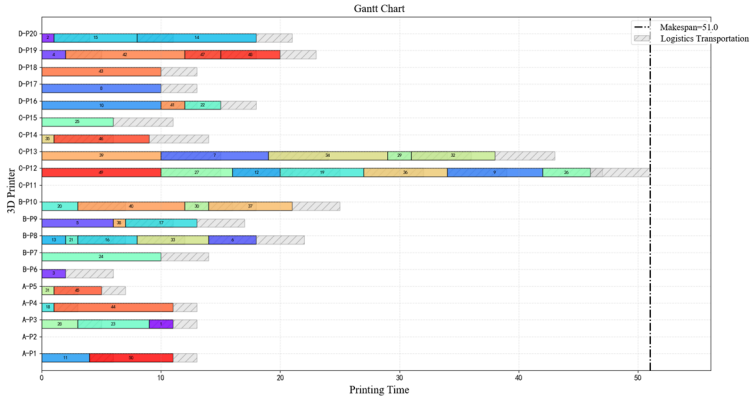

Fig 2. Initial scheduling solution under machine failure conditions

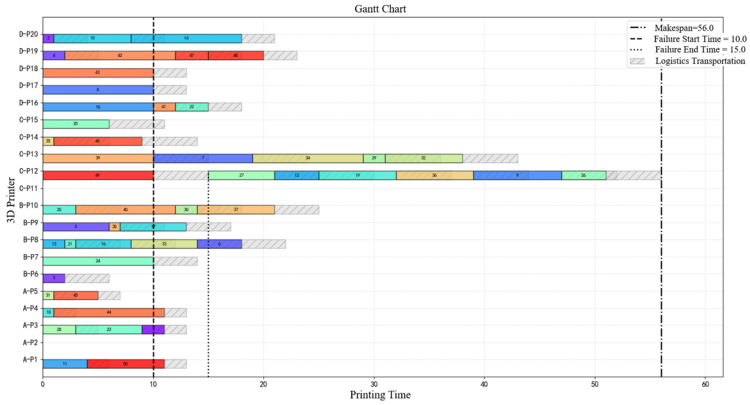

Fig 3. Right-shift rescheduling solution under machine failure conditions

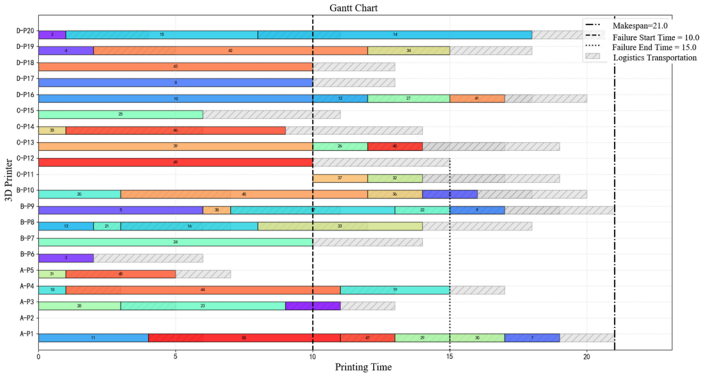

Fig 4. Full rescheduling solution under machine failure conditions

ISSN: 2688-8645

Figure 5 illustrates completion time convergence under machine failure. The algorithm converges rapidly from 30, stabilizing at 21 around the 18th generation. The curve remains flat until the 100th generation without local optima, indicating a global stable state. This confirms the model effectively reduces capacity loss via flexible allocation.

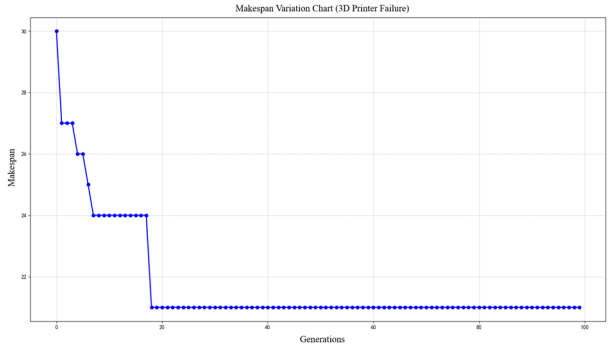

Fig 5. Graph of makespan changes under machine failure conditions

Figure 6 shows processing cost changes under machine failure. The downward trend indicates the full rescheduling strategy successfully optimizes total production cost. Step-like fluctuations reflect crossover and mutation operators maintaining diversity to break local optima. A significant cost reduction occurs around generation 40, stabilizing below 3000, indicating the algorithm found the optimal cost balance point. The sharp initial drop and subsequent stabilization demonstrate the model's quick response and ability to find economically superior schemes in resource-constrained failure scenarios.

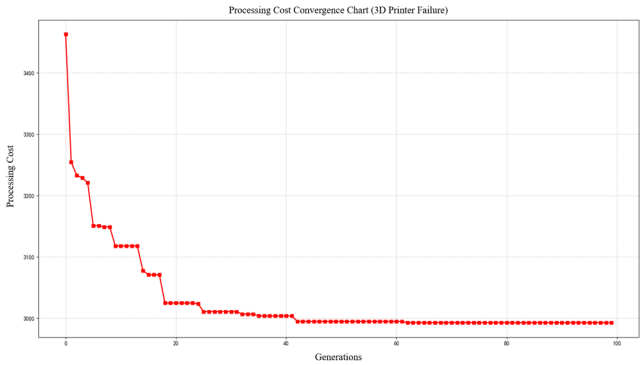

Fig 6. Graph of processing cost changes under machine failure conditions

The right-shift rescheduling strategy retains the original machine allocation scheme, so its processing cost and logistics cost remain consistent with the initial scheduling scheme, and it cannot avoid the time loss caused by the failure, resulting in a significant extension of the completion time. In contrast, the full rescheduling strategy calculates the actual processing cost of each workpiece on different available machines through the greedy mutation operator in the code, and tends to assign workpieces to those machines with lower costs, thereby achieving a decrease in total production cost at the macro level.

5. Conclusion and Outlook

This paper addresses dynamic disturbances in distributed 3D printing, specifically equipment failure. Key contributions include: (1) constructing a multi-objective event-driven model balancing makespan and cost; (2) proposing an improved NSGA-II algorithm with chromosome segmentation and adaptive operators; (3) verifying via simulation that the method outperforms traditional right-shift rescheduling in efficiency and cost. Future work will expand disturbance

types (e.g., logistics, energy) and integrate deep reinforcement learning for active prediction. This research supports high-flexibility additive manufacturing in Industry 4.0.

References

[1] Zhai Y, Lados DA, LaGoy JL. Additive manufacturing: Making imagination the major limitation. JOM 2014; 66:808–16.

[2] Zhang J, Ding G, Zou Y, et al. Review of job shop scheduling research and its new perspectives under industry 4.0. J Intell Manuf 2019; 30:1809–30.

[3] Zeng L, Shi J, Li Y, et al. A strengthened dominance relation NSGA-III algorithm based on differential evolution to solve job shop scheduling problem. Comput Mater Contin 2024; 78:375–92.

[4] Ouyang HC, Zhang TR, Wu DH. Multi-objective Flexible Job Shop Scheduling Based on Improved NSGA-III Algorithm. Control Engineering of China 2023;30:105–12.

[5] Li XH, Wang XR, Zhao Y, et al. Dynamic Scheduling in Cloud Manufacturing Environment. Computer Systems & Applications 2021; 30:225–31.

[6] Darwish LR, El-Wakad MT, Farag MM. Towards sustainable industry 4.0: A green real-time IIoT multitask scheduling architecture for distributed 3D printing services. J Manuf Syst 2021; 61:196–209.

[7] He J, Wu J, Siau KL. Task scheduling strategy for 3DPCP considering multidynamic information perturbation in green scene: J Glob Inf Manag 2024; 32: 1–23.

[8] Poudel L, Zhou W, Sha Z. Resource-constrained scheduling for multi-robot cooperative three-dimensional printing. J Mech Des 2021; 143: 072002.

[9] He J, Wu J, Ni J, et al. Real-time task scheduling strategy for 3D printing cloud platforms in health scenes. Appl Intell 2025; 55:1002.

[10] Gao K, Yang F, Zhou M, et al. Flexible job-shop rescheduling for new job insertion by using discrete jaya algorithm. IEEE Trans Cybern 2019; 49: 1944–55.

[11] Stevenson Z, Fukasawa R, Ricardez-Sandoval L. Evaluating periodic rescheduling policies using a rolling horizon framework in an industrial-scale multipurpose plant. J Sched 2020; 23:397–410.

[12] Mejía G, Montoya C, Bolívar S, et al. Job shop rescheduling with rework and reconditioning in industry 4.0: An event-driven approach. Int J Adv Manuf Technol 2022; 119: 3729–45.