Industrial Engineering and Innovation Management (2024)
Clausius Scientific Press, Canada

DOI: 10.23977/ieim.2024.070408
ISSN 2522-6924 Vol. 7 Num. 4

Research on Multi-stage Decision Optimization Models in the Production Process

Yutong Zhang

School of Electronics and Information Technology (School of Microelectronics), Sun Yat-Sen University, Guangzhou, 510006, China

Keywords: Statistical Testing, Integer Programming, Genetic Algorithms

Abstract: Product quality is crucial for a company's competitive edge. In the assembly process, a single defective part can render the entire product substandard; even with all parts in order, the finished product may still be defective. For non-conforming products, companies can choose to scrap or disassemble them, incurring costs but recovering parts. Additionally, companies bear the costs of replacing products returned due to quality issues. This article analyzes and models multi-stage decision-making in product manufacturing, aiming to provide effective solutions. By integrating optimization modeling with statistical and managerial theories, including dynamic programming, integer programming, branch and bound methods, and genetic algorithms, it addresses various scenarios faced by companies during production: from part procurement and testing to assembly, and from product sales to the handling of substandard products. An efficient, scientific, and economical testing and production decision-making plan is designed to maximize profits or minimize costs by optimizing these decision points. This not only helps companies improve product quality, reduce production and rework costs, but also enhances their market competitiveness.

1. Introduction

In the modern manufacturing industry, product quality is one of the core factors for enterprises to maintain competitiveness. The production process relies on a variety of key components. During the assembly process, if one of the components has a defect, the entire finished product will be directly judged as unqualified; even if all components are qualified, the finished product may not pass the inspection. For unqualified finished products, companies can choose to scrap or dismantle. The dismantling process will not damage the components, but it will generate additional costs. In addition, the company promises to unconditionally replace products returned due to quality issues, and bear the logistics costs and reputation losses incurred. Against this background, how to scientifically and reasonably formulate testing and decision-making plans has become an urgent challenge for companies to solve.

In past research, many scholars have been committed to establishing different optimization models to solve decision-making problems in the production process. For example, Chen L and others [1] proposed a plant growth simulation algorithm to solve integer programming problems, which searches for the global optimal solution by simulating the growth process of plants. Gad A G [2] used

particle swarm optimization to solve integer programming problems, improving search efficiency through collective intelligence. In addition, genetic algorithms, as an effective global optimization method, have been widely used in production decision optimization. Lambora A and others [3] reviewed genetic algorithms, pointing out their advantages in dealing with complex optimization problems. Chen J and others [4] studied improved methods of genetic algorithms, increasing the convergence speed and solution quality of the algorithms. These studies have provided strong tools and theoretical foundations for production decision optimization.

Despite the progress made in the field of production decision optimization, there are still some shortcomings. First, most studies focus on a single production stage and do not fully consider the complexity of multi-stage decision-making in the production process. Second, existing models often ignore the impact of uncertainties in the production process, such as market demand fluctuations and the instability of raw material supply. In addition, some models rely too much on precise mathematical expressions, neglecting the operability and flexibility in the actual production process. In response to these shortcomings, this paper proposes a new method based on a multi-stage decision optimization model. This study not only considers multiple stages in the production process but also introduces a combination of dynamic programming and integer programming methods to more comprehensively capture decision points in the production process. At the same time, this paper introduces genetic algorithms to deal with uncertainties in the production process, enhancing the model's adaptability and flexibility. In addition, this paper also verifies the effectiveness and practicality of the proposed model through actual case analysis, providing enterprises with a scientific, economical, and efficient production decision support tool.

2. The basic fundamental of Multi-stage Decision Optimization Model

2.1 Optimal Strategy Model Based on Integer Programming

Imagine a company that encounters six different production scenarios during its manufacturing process due to differences in 'defective rates, purchase prices, and inspection costs of Components 1 and 2, defective rates, assembly costs, inspection costs, market selling prices, replacement costs for non-conforming finished products, and disassembly costs of the finished products'.

This paper employs an integer programming solution strategy - the branch and bound method to address the optimal cost-benefit problem. Initially, an analysis of the applicability of integer programming and decision-making is conducted. The goal of the model is to identify the best combination of testing and processing strategies to minimize total costs. The flowchart of the optimal strategy for integer programming is shown in Figure 1 below.

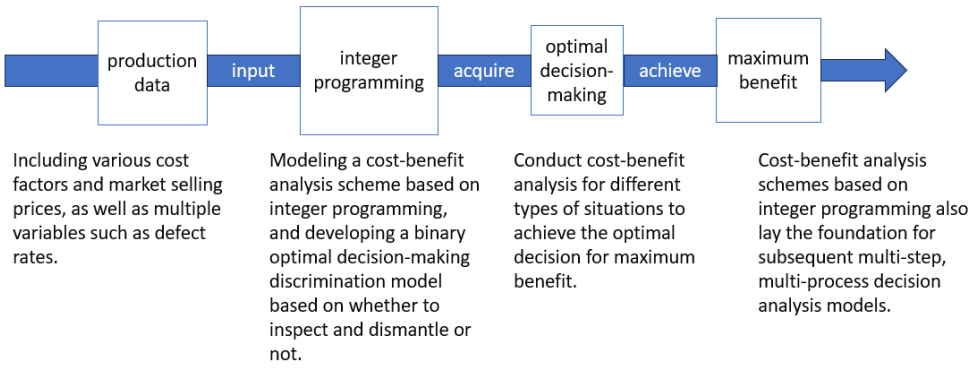


Figure 1: Optimal Strategy Modeling Diagram Based on Integer Programming

When the problem is relatively simple and involves fewer objectives, the branch and bound (B&B)

method is generally used to obtain the exact solution of integer programming. The core idea is to decompose this NP-hard problem into solving a series of linear programming (LP) problems (each LP problem is solvable in polynomial time), and track the upper bound (optimal feasible solution) and lower bound (optimal linear relaxation solution) of the original problem in real-time during the solution process. After branching, bounding is required, that is, a feasible domain range must be determined. Since the decision tree grows exponentially, for example, a 4-level decision tree will have 16 planning problems, which greatly increases the computational burden. Therefore, it is necessary to specify a feasible range to reduce computational costs and obtain the optimal decision, such as inspecting or not inspecting components, more quickly. The selection of the feasible domain includes feasibility pruning (see Figure 2), that is, if the bound value of a sub-problem has already exceeded the currently known optimal solution, then this sub-problem can be pruned, that is, no further exploration is needed; optimality pruning, that is, if the optimal solution of a sub-problem cannot be better than the currently known optimal solution, then this sub-problem can also be pruned.

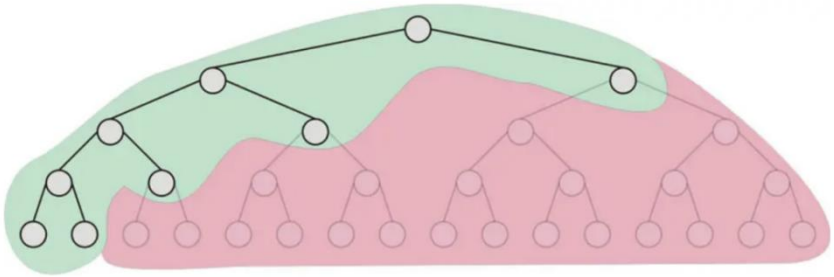


Figure 2: Branch and Bound Illustration of Variable Decision Tree

Since each component has a certain defect rate, the quality of each component will directly affect the overall quality of the final product. Therefore, the company needs to make optimal key decisions at multiple stages of production based on the known defect rates of components and finished products, such as whether to inspect components, whether to inspect finished products, and whether to dismantle unqualified finished products for recycling usable components, in order to maximize profit or minimize costs. This paper is based on dynamic programming and integer programming optimization models, by defining decision variables, establishing objective functions and constraints, and using sensitivity analysis to evaluate the economic benefits of different decision-making schemes. The rationality of this method lies in the system's ability to quantify the impact of various decisions on total costs and total revenue, providing data-driven decision support for enterprises [5] , ensuring that while meeting production needs and ensuring product quality, the total cost of production is minimized as much as possible, and cost-effectiveness is maximized.

First, consider establishing a unified profit objective function for different situations. Profit is mainly composed of the following three parts:

1) Direct costs: Including explicit cost expenditures such as purchase costs, inspection costs, assembly, and dismantling costs.

2) Replacement loss: Replacement loss caused by unqualified products that have not been inspected.

3) Revenue: Revenue brought by the sale of qualified products.

Set decision variables. From the above analysis, it is known to be a binary decision problem, thus there are the following decision variables:

x1, x2: Whether to inspect components 1 and 2 (0 or 1)

y: Whether to inspect finished products (0 or 1)

z: Whether to dismantle unqualified finished products (0 or 1)

Finally, consider the objective function to maximize profit, thus it has the following formula:

$$\begin{aligned}&\text{Maximize } P = sy\left(1-p_1\left(1-x_1\right)-p_2\left(1-x_2\right)\right)-c_1x_1-c_2x_2\\&-ay-d_1x_1-d_2x_2-dy-l(1-y)\left(p_1\left(1-x_1\right)+p_2\left(1-x_2\right)\right)-rz\end{aligned} \tag{1}$$

where $p1$, $p2$ are the defect rates of Component 1 and Component 2, $c1$,$c2$ represent the purchase prices of Component 1 and Component 2, $d1$,$d2$ mean the inspection costs of Component 1 and Component 2, a is the assembly cost, dd is the inspection cost of the finished product, s means the market selling price, l represents the replacement loss, and r is the dismantling cost.

The constraints include:

1) Component inspection constraints: The decision set is an integer, either inspect or not inspect, $x_i \in \{0,1\}$, for all i=1,2.

2) Semi-finished product inspection constraints: The decision set is an integer, ensuring that semi-finished products at each process are either inspected or not inspected, $y_j \in \{0,1\}$, for all j=1,2.

3) Dismantling decision constraints: If the semi-finished product is inspected and found to be unqualified, then decide whether to dismantle, $z \le 1-y_j$, for all j=1,2, $z \in \{0,1\}$.

The optimization problem and its constraints are:

$$\begin{aligned}&\text{Maximize } P = sy\left(1-p_1\left(1-x_1\right)-p_2\left(1-x_2\right)\right)-c_1x_1-c_2x_2\\&-ay-d_1x_1-d_2x_2-dy-l(1-y)\left(p_1\left(1-x_1\right)+p_2\left(1-x_2\right)\right)-rz\\&s.t.\ \ x_i \in \{0,1\}, \forall i = 1,2\\&y_j \in \{0,1\}, \forall j = 1,2\\&z \le 1-y_j, \forall j = 1,2, z \in \{0,1\}\end{aligned} \tag{2}$$

This paper considers solving the problem using the branch and bound method based on Matlab, and the following Figure 3 is the flowchart of the code:

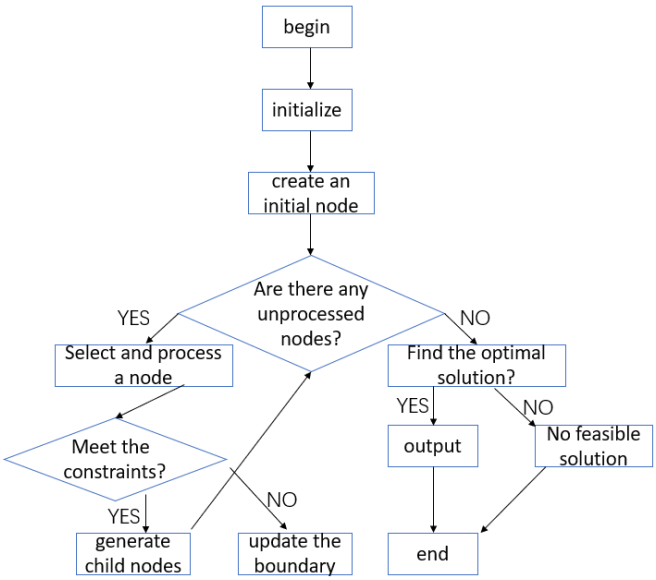


Figure 3: Branch and Bound Algorithm Flowchart

2.2 Genetic Algorithm

In light of the aforementioned issues, considering the interactions and cumulative effects of multiple components and semi-finished products across various processes, along with different defect rates, inspection costs, assembly costs, and dismantling fees for m processes and n components, semi-

finished products, and finished products, significantly increases the dimensions and complexity of optimization decisions. To address this challenge, this paper plans to decompose the inspection, assembly, and dismantling steps in each process step by step, using stochastic programming and multi-objective optimization methods to construct a multi-stage decision model. This model takes into account all key factors in the production process, including multiple components and processes, to ensure that while meeting quality requirements, costs are optimized and production efficiency is improved. The model can adapt to uncertainties and changes in the production process, providing specific decision support and helping enterprises make scientific and reasonable decisions in actual operations [6], thereby finding the optimal balance point to maximize cost-effectiveness while ensuring product quality.

Since the multi-stage optimal decision-making problem involves a set of parameters, using precise solutions such as branch and bound will be too costly and is not conducive to improving enterprise efficiency. Therefore, heuristic methods are considered to ensure accuracy while enhancing the efficiency of the solution process. Specifically, a genetic algorithm is considered, which involves a more extensive parameter decision space. Genetic algorithms improve the model's adaptability and flexibility when dealing with uncertainties and multi-stage decision-making problems, enhancing the optimization capabilities for complex production systems.

Genetic algorithms are heuristic search algorithms that simulate the evolutionary process in nature. In the context of assembly production, especially in complex production environments involving multiple processes and components, genetic algorithms provide an effective optimization tool that efficiently searches a wide range of solution spaces. They not only improve search efficiency but also enhance the exploration capabilities for the global optimal solution. This paper establishes a genetic algorithm-driven multi-stage integer programming model to solve the aforementioned optimal decision-making problem for m processes and n components.

Specifically, solving the multi-stage optimal decision-making problem can be translated into the following steps in the genetic algorithm:

Step 1: Initialize the population: Randomly generate a set of candidate solutions as the initial population. In this part, it is the selection of production strategies, i.e., whether to inspect components and whether to dismantle parts. They can be regarded as the final results to be optimized.

Step 2: Fitness function evaluation: Calculate the fitness of each individual (solution), where fitness represents the optimization objective function in this part.

Step 3: Selection operation: Select individuals from the current population based on fitness for crossover and mutation operations. In assembly production, this corresponds to screening out better-performing production plans.

Step 4: Crossover and mutation operations: Cross and mutate some genes of individuals with a certain probability to increase the diversity of the population. In assembly production, this corresponds to exploring more effective production methods, avoiding local optima, and increasing the chances of finding the global optimal solution.

Let the quantity of finished products be *n*, and the fitness function be *fi*. Then the probability *Pi* that it is selected is given by:

$$p_i = \frac{f_i}{\sum_{k=1}^{n} f_k} \tag{3}$$

Probability of selecting the i-th individual:

$$p_i = q^{'}(1-q)^{r-1} \tag{4}$$

The formula for transforming gene values is:

$$x_i' = \begin{cases} x_i & i < r \\ y_i & otherwise \end{cases}$$
$$y_i' = \begin{cases} y_i & i < r \\ x_i & otherwise \end{cases} \quad (5)$$

The mutation formula is:

$$x_i' = \begin{cases} x_i + (b_i - x_i)f(G), i = j, r_1 < 0.5 \\ x_i - (x_i + a_i)f(G), i = j, r_1 \leq 0.5 \\ x_i \quad otherwise \end{cases} \quad (6)$$

$f(G) = (r_2(1 - \frac{G}{G_{\max}}))^b$, $r_1, r_2 = U(0,1)$, G represents the current number of generations, G_{max} is the maximum number of generations, the objective function value $f(x)$ is transformed into the fitness function value $Fit(f(x))$, let:

$$Fit(f(x)) = \begin{cases} f(x) & \text{The objective function is the maximization function} \\ -f(x) & \text{The objective function is the minimization function} \end{cases} \quad (7)$$

For optimization problems, they can be transformed into:

$$min \ f(x)$$
$$s.t. \begin{cases} g_i(x) \leq 0; i = 1,2 \ldots m \\ x \in X \\ Number\ of\ parameters\ x = n - l \end{cases} \quad (8)$$

The state variable x' can be directly solved by $h_j(x') = 0; j = 1,2 \ldots l$.

3. Results

3.1 Solving the Optimal Strategy Model Based on Integer Programming

For the convenience of solving the model, a constraint diagram of integer programming is shown in Figure 4. The blue line represents a linear inequality, but here the x and y independent variables are constrained to integer variables, so the feasible region becomes several discrete black dots within the red line area(If it is linear programming, the feasible region is all the areas inside the blue line segment).

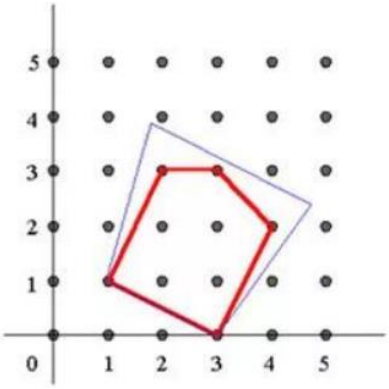
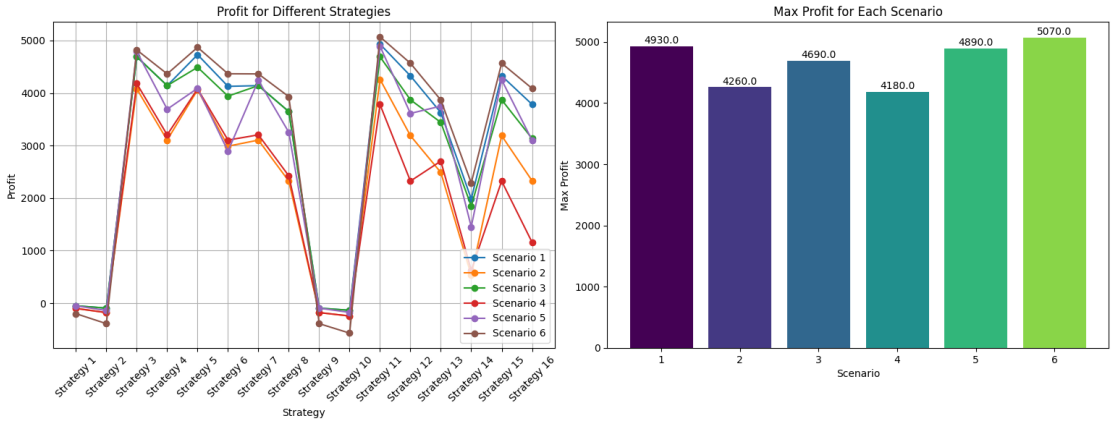


Figure 4: Discrete Feasible Region Diagram of Integer Programming

Taking Situation 1 as an example, the optimal decision obtained by Matlab is $[x_1, x_2, y, z]=[0,1,1,1]$, which means Component 1 is not inspected, Component 2 is inspected, the finished product is inspected, and unqualified finished products are dismantled.

The cost-benefit visualization graph is shown in Figure 5:



(a) Profit for Different Strategies (b) Max Profit for Each Scenario

Figure 5: Cost-benefit visualization graph

In order to eliminate the influence of multifactorial effects, a univariate analysis is attempted. Based on the branch and bound parameters determined from the previous analysis, a model is established and simulated in the Matlab environment. This model can be used to analyze the interrelationships between factors. Finally, residual white noise testing is conducted, and it is observed from Figure 6 that after differentiation, the majority of correlation coefficients fall within the confidence interval, thus confirming the residual terms of the fitted model to be a white noise sequence, which can be used for subsequent calculations.

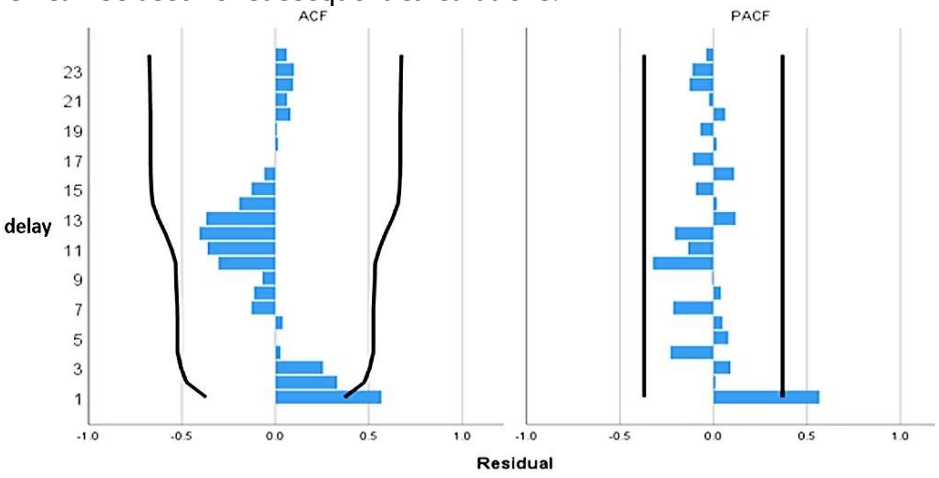


Figure 6: Residual Test Diagram

3.2 Solving with Genetic Algorithm

This section aims to solve the multi-stage optimal decision-making problem for m processes and n components. A problem with 1 process and 2 components can be regarded as a basic decision model. Therefore, m processes can be considered as an n-stage decision, each stage including n components, and decisions are made on whether to inspect them. Thus, there will be no fewer than m+n variables for the optimal decision.

Define decision variables:

• *xi*: Whether to inspect component i, *i*=1,2,…,*n*. 1 indicates inspection, 0 indicates no inspection.

• *yj*: Whether to inspect the semi-finished product after the j-th process, *j*=1,2,…,*m*. 1 indicates inspection, 0 indicates no inspection.

• *zj*: Whether to dismantle the unqualified semi-finished product from the j-th process, *j*=1,2,…,*m*. 1 indicates dismantling, 0 indicates no dismantling.

The task for m processes and n components is to solve for maximum profit:

$$\text{Maximize } P = s\left(1-\sum_{i=1}^{n} p_i(1-x_i)\right)-\sum_{i=1}^{n}(c_i+d_i x_i)-\sum_{j=1}^{m}\left(a_j y_j+d_j y_j+r_j z_j+l\left(1-y_j\right)\right) \tag{9}$$

where *pi* is the defect rate of the *i*-th component, *ci* represents the purchase price of the *i*-th component, *di* means the cost of inspecting the *i*-th component, *aj* is the assembly cost for the *j*-th process, *dj* is the cost of inspecting the semi-finished product of the *j*-th process, *s* means the market selling price of the final product, *l* represents the replacement loss of unqualified products, and *rj* is the cost of dismantling the unqualified semi-finished product from the *j*-th process.

The constraints include:

Component Inspection Constraint: The decision set is an integer, ensuring that each component is either inspected or not inspected. $x_i \in \{0,1\}$, for all $i=1,\ldots,n$.

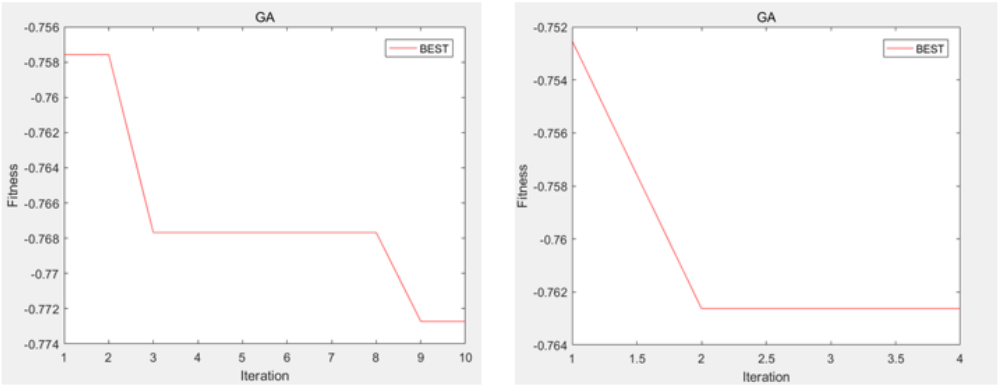
Semi-finished Product Inspection Constraint: The decision set is an integer, ensuring that the semi-finished product of each process is either inspected or not inspected. $y_j \in \{0,1\}$, for all $j=1,\ldots,m$.

Dismantling Decision Constraint: If the semi-finished product is inspected and found to be unqualified, then decide whether to dismantle. $z_j \leq 1-y_j$, for all $j=1,\ldots,m$.

In summary, the multi-stage optimization problem and constraints are:

$$\begin{aligned} \text{Maximize } P &= s\left(1-\sum_{i=1}^{n} p_i(1-x_i)\right)-\sum_{i=1}^{n}(c_i+d_i x_i)-\sum_{j=1}^{m}\left(a_j y_j+d_j y_j+r_j z_j+l\left(1-y_j\right)\right) \\ s.t.\ & x_i \in \{0,1\}, \forall i=1,\ldots,n \\ & y_j \in \{0,1\}, \forall j=1,\ldots,m \\ & z_j \leq 1-y_j, \forall j=1,\ldots,m \end{aligned} \tag{10}$$

Through optimization analysis, the current optimization result can be obtained as shown in Figure 7 below:



(a) Optimization is performed 100 times (b) Optimization is performed 500 times

Figure 7: The genetic algorithm optimization is performed with different execution times

By using the established genetic algorithm-driven optimal decision model to solve a specific case, where *m*=2 and *n*=8. Here, only the function call for a specific situation is shown. In an already determined situation, people can directly call this genetic algorithm model to find the optimized

production strategy. Schematic diagram of genetic algorithm search force optimization convergence is presented in Figure 8.

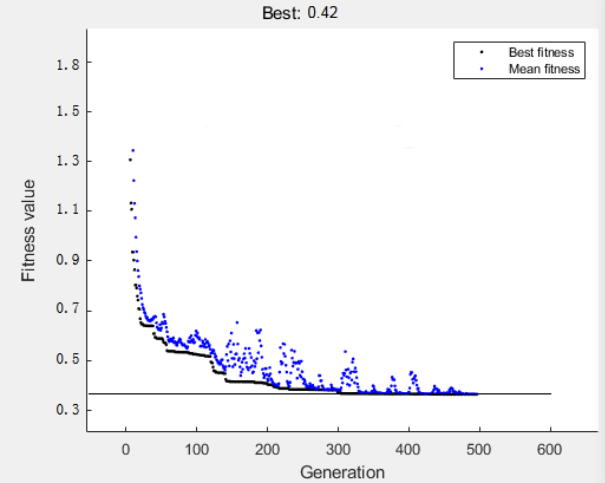


Figure 8: Schematic diagram of genetic algorithm search force optimization convergence

According to the results of the program call, in the production process of 2 processes and 8 components, the optimal decision for maximizing production profit is (1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1), which corresponds to whether to inspect each of the 8 components, whether to inspect the 3 semi-finished products, whether to inspect the finished product, and whether to dismantle the unqualified product, respectively.

4. Conclusions and outlooks

This paper presents an in-depth analysis and modeling of the multi-stage decision-making challenges in the production process of electronic products, offering an effective solution. By integrating optimization techniques with statistical and managerial methods, such as dynamic programming, integer programming, branch and bound methods, and genetic algorithms, the model addresses key production issues including component procurement, inspection, assembly, market sales of finished products, and handling of defective items. The model is designed to maximize profits or minimize costs, thereby improving product quality, reducing production and rework expenses, and enhancing market competitiveness.

Its comprehensive, flexible, and risk-aware approach also makes it applicable to other manufacturing sectors like automotive, machinery, and aviation, providing a practical and economically beneficial decision-making framework.

References

[1] Chen L, Liu Q, Ye C, et al. A novel decision-making scheme for hospital emergency services based on plant growth simulation algorithm [J]. International Journal of Internet Manufacturing and Services, 2024, 10(2-3): 112-131.

[2] Gad A G. Particle swarm optimization algorithm and its applications: a systematic review[J]. Archives of computational methods in engineering, 2022, 29(5): 2531-2561.

[3] Lambora A, Gupta K, Chopra K. Genetic algorithm-A literature review[C]//2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon). IEEE, 2019: 380-384.

[4] Chen J, Zhao F, Sun Y, et al. Improved XGBoost model based on genetic algorithm[J]. International Journal of Computer Applications in Technology, 2020, 62(3): 240-245.

[5] Ding Y , Shen G , Wan W .Research on a Multi-Objective Optimization Method for Transient Flow Oscillation in Multi-Stage Pressurized Pump Stations[J].Water (20734441), 2024, 16(12).DOI:10.3390/w16121728.

[6] Kim J H, Lee Y, Kim W C, et al. Goal-based investing based on multi-stage robust portfolio optimization[J]. Annals of Operations Research, 2022, 313(2): 1141-1158.