APEM journal
Advances in Production Engineering & Management
Volume 16 | Number 2 | June 2021 | pp 240–252
https://doi.org/10.14743/apem2021.2.397
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Optimization of disassembly line balancing using an improved multi-objective Genetic Algorithm

Wang, Y.J.[a,*], Wang, N.D.[a], Cheng, S.M.[a], Zhang, X.C.[a], Liu, H.Y.[a], Shi, J.L.[a], Ma, Q.Y.[a], Zhou, M.J.[a]

[a]School of Mechanical Engineering and Automation, Dalian Polytechnic University, P.R. China

ABSTRACT

Disassembly activities take place in various recovery operations including remanufacturing, recycling, and disposal. Product disassembly is an effective way to recycle waste products, and it is a necessary condition to make the product life cycle complete. According to the characteristics of the product disassembly line, based on minimizing the number of workstations and balancing the idle time in the station, the harmful index, the demand index, and the number of direction changes are proposed as new optimization objectives. So based on the analysis of the traditional genetic algorithm into the precocious phenomenon, this paper constructed the multi-objective relationship of the disassembly line balance problem. The disassembly line balance problem belongs to the NP-hard problem, and the intelligent optimization algorithm shows excellent performance in solving this problem. Considering the characteristics of the traditional method solving the multi-objective disassembly line balance problem that the solution result was single and could not meet many objectives of balance, a multi-objective improved genetic algorithm was proposed to solve the model. The algorithm speeds up the convergence speed of the algorithm. Based on the example of the basic disassembly task, by comparing with the existing single objective heuristic algorithm, the multi-objective improved genetic algorithm was verified to be effective and feasible, and it was applied to the actual disassembly example to obtain the balance optimization scheme. Two case studies are given: a disassembly process of the automobile engine and a disassembly of the computer components.

ARTICLE INFO

Keywords:
Assembly;
Disassembly;
Line balancing;
Multi-objective optimization;
Remanufacturing;
Product recovery;
Product life cycle;
NP-hard problem;
Improved genetic algorithm

**Corresponding author:*
wangyj@dlpu.edu.cn
(Wang, Y.J.)

Article history:
Received 13 May 2021
Revised 23 May 2021
Accepted 4 June 2021

1. Introduction

With the continuous development of science and technology, the demand for new products is increasing, and the number of waste products and components is increasing, which will inevitably cause pollution to the environment. In order to solve the shortage of resources and realize sustainable development, the enterprise must pay attention to the recycling and reuse of waste products. Disassembly is the basic action of product recycling, extracting useful parts and recycling harmful parts to achieve a circular economy and green manufacturing.

In recent years, more and more researchers have devoted themselves to the disassembly line balancing problem (DLBP). Gungor and Gupta [1] analyzed and described the DLBP problem and put forward the influencing factors. Avikal *et al*. [2] used heuristic algorithm to solve DLBP problem, but there are some limitations. Altekin and Akkan [3] used a linear programming method to optimize DLBP to achieve the purpose of a balanced disassembly line. The Genetic Algorithm was used to solve the disassembly line balancing problem and the optimal solution was obtained

in an effective time. Kailayci *et al.* adopted a simulated annealing algorithm, which had strong processing capacity and better local search capability than other algorithms. However, it took a long time and had a weak ability to obtain a globally optimal solution [4, 5]. Nikola *et al.* solved the DLBP problem under multi-objective conditions in actual production [6-10]. Ding *et al.* [11] proposed an ant colony algorithm based on Pareto to optimize the four objectives. Cao *et al.* solved disassembly line balancing problems with different algorithms [12-22].

By analyzing the above works of literature and methods, it is easy to see intelligent optimization algorithms such as Genetic Algorithm, which have excellent performance in solving multi-objective optimization problems.

Given the shortcomings of the above researches, an improved genetic algorithm for solving the disassembly line balancing problem was proposed, and its advantages for solving this kind of problem were analyzed and verified through specific problems and examples.

The paper is organized as follows. The mathematical model is summarized in Section 2. Optimization and analysis with multiple objectives are also described in this section. Section 3 is the presentation of the solution. The case analysis and discussion are reported in Section 4. Finally, the conclusions are reported in Section 5.

2. Mathematical models

2.1 Basic assumptions

There are many uncertainties in the actual disassembly process, and these uncertainties are bound to affect the production beat of the disassembly online operation. Considering the application scope of the model in practice, this paper ignores extreme phenomena in the process of building the U-shaped DLBP model, so the following assumptions are made.

1) The production beat is unchanged.
2) The disassembly resources are the same or similar in structure.
3) The resource is completely disassembled, and the operation stops after the disassembly to the last part.
4) All disassembly tasks are performed on the disassembly line with no missing parts during the operation.
5) Operator proficiency and experience are the same, that is, the disassembly time of each part does not vary from operator to operator
6) During the disassembly, the required parts are guaranteed to be intact, that is, the disassembly process can bring economic benefits.
7) The disassembly time of the parts is not affected by external factors such as product quality, and there is no dependence between the parts.
8) The disassembly process is carried out according to the disassembly task and cannot be disassembled.
9) Operating time is normally distributed ′Y$_{ش}$ ~ ŭ ‘, „ ٮ .
10) The constraints are as follows.
    a) The total operation time of each workstation shall not exceed the production beat.
    b) The same disassembly operation shall not be carried out in two or more workstations.

2.2 Basic parameters

Assuming that each unassembled part is a disassembly task, the number of parts is è (also known as the number of disassembly tasks). ′Y is the set of all the tasks, ′Y = 1,2,3, … , è . ŭ is the number of workstations. ő′Y is the production beat. The parameter ȯ$_{ش}$ represents the sum of activity time of all disassembly tasks allocated on the workstation ٪.

2.3 Decision variables

The variable x$_{ش}$ represents the relationship between task ٫ and workstation ٪, ώ$_{ش}$ هي 1, otherwise ώ$_{ش}$ هي 0.

$$\text{ω}_{\text{ش س}} \text{ هي } \begin{matrix} 1, & \text{task } \text{ا} \text{ is assigned to workstation } \text{ك} \\ 0, & \text{else} \end{matrix}$$

The balanced optimization of the disassembly line should not only consider the balanced distribution of work tasks but also include the environment and resources. The products to be dismantled may contain harmful substances, such as heavy metals and chemical poisons, which should be given priority in the disassembly operation. The main purpose of disassembly is to recycle and use the spare parts with surplus value, and the valuable parts should be dismantled as soon as possible. To minimize the disassembly time of the product to be disassembled, the number of changes of disassembly orientation is also included in the optimization space to shorten the disassembly time.

In this paper, five objectives of disassembly line balancing are considered and optimized.

1) Minimum number of workstations.
2) Balance the free time of each workstation.
3) Disassemble high-demand components as soon as possible.
4) Disassemble hazardous parts as soon as possible.
5) The least change of direction for disassembling.

$$\text{ا}\text{ك} \text{ هي } \text{ك} \tag{1}$$

$$\text{ا}\text{ك}_{\text{ب}}=\sum_{\text{ش}=1}^{\text{ج}} \text{ض}_{\text{ش}} \text{ سر } \text{ب} \tag{2}$$

$$\text{ك}_{\text{ب}} \text{ هي } \text{ضحي } \sum_{\text{ش}=\text{ب}}^{\text{ج}} \text{ض}_{\text{ش}} \text{ سر } \text{ب} \tag{3}$$

$$\text{ا}\text{ك}_{\text{ب}} \text{ هي } \text{ضحي } \sum_{\text{ش}=\text{ب}}^{\text{ج}} \text{ض}_{\text{ش}} \text{ سر } \text{ب} \tag{4}$$

$$\text{ا}\text{ك}_{\text{ن}} \text{ هي } \text{ضحي } \text{ا}\text{ه}_{\text{حر}} \text{ سر } \sum_{\text{ش}=\text{ب}}^{\text{ج}} \tag{5}$$

$$\text{ا}\text{ك}_{\text{ع}} \text{ هي } \text{ضحي } \text{ا}\text{ق}_{\text{حر}} \text{ سر } \sum_{\text{ش}=\text{ب}}^{\text{ج}} \tag{6}$$

where $\text{ه}_{\text{حر}}$ هي $\begin{matrix} 1 \text{ needed} \\ 0 \quad \text{else} \end{matrix}$, $\text{ق}_{\text{حر}}$ هي $\begin{matrix} 1 \text{ harmful} \\ 0 \quad \text{else} \end{matrix}$.

The direction indicators are introduced to evaluate the solution sequence. The smaller the value, the less the change of direction in the disassembly process, the better the solution will be. The following relation is used to represent each direction of the disassembly process relative to the parts and workstations.

$$\text{ك}_{\text{حر}} \text{ هي } \begin{matrix} \text{سر } 3 & \text{سر Z direction} \\ \text{سر } 2 & \text{سر Y direction} \\ \text{سر } 1 & \text{سر X direction} \\ 1 & \text{شر X direction} \\ 2 & \text{شر Y direction} \\ 3 & \text{شر Z direction} \end{matrix} \tag{7}$$

The direction indicators are expressed in the form of decision variables.

$$\text{ا}\text{ك}_{\text{ن}} \text{ هي } \sum_{\text{ش}=\text{ب}}^{\text{ج}} \text{ك}_{\text{ش}}, \text{ك}_{\text{س}} \text{ هي } \begin{matrix} 1, \text{ك}_{\text{حر}} \neq \text{ك}_{\text{حر}}, \\ 0, \text{ك}_{\text{حر}} \text{ هي } \text{ك}_{\text{حر}}, \end{matrix} \tag{8}$$

The multi-objective disassembly line balance is represented as the following model by the above balance objectives.

$$\min F = (f_1, f_2, f_3, f_4) \tag{9}$$

s.t.

$$\sum_{j=1}^{n} t_j / CT \le m \le n \tag{10}$$

$$\sum_{j=1}^{n} x_{kj} t_j \le CT, \quad k = 1,2,\ldots,m \tag{11}$$

$$\sum_{k=1}^{n} k\, x_{kp} \le \sum_{k=1}^{n} k\, x_{ks}, \quad \forall p, k_{ps} = 1 \tag{12}$$

The objective function (Eq. 1) represents the minimum number of workstations. The objective function (Eq. 2) means to balance the free time of each workstation. The objective function (Eq. 5) represents the index value of disassembling parts with high demand first. The objective function (Eq. 6) represents the index value of priority disassembly of hazardous parts. The objective function (Eq. 8) represents the least change of direction for disassembling.

The constraint function (Eq. 10) means that the number of workstations should be no less than the number of theoretical workstations and no more than the number of disassembly tasks. The constraint function (Eq. 11) represents that the disassembly time in each workstation shall not exceed the production beat. The constraint function (Eq. 12) means the disassembly sequence must meet the disassembly priority relationship.

3. Presentation of solution and used genetic operations

Genetic Algorithm (GA) was proposed in 1967 by a scientific research team led by John Holland of the University of Michigan [23]. It is a natural evolution-based algorithm for intelligent optimization search based on the concept of biological evolution and the laws of biogenetics and natural selection. GA relies on the information exchange of the individuals in the community and population search, to encode the parameters of the solution as genes, and several genes constitute a chromosome, as an individual, many chromosomes experience generation genetic by crossover, selection and mutation operation, the search results gradually converge to the region where the optimal solution is located until the optimal solution is found. The advantage of GA in solving the problem of activity order optimization is that the optimal solution can be found without going through the space of all solutions, and the solution effect for the multi-objective optimization problem is significant.

3.1 Coding

Encoding methods include binary encoding, float-point encoding, real number encoding, and so on. Given the disassembly line activity tasks, by using a chromosome encoding rules based on activity order successively, *n* disassembly elements are arranged in a row, corresponding to a gene site, and according to the priority diagram of activity order, these disassembly elements are assigned to each workstation and are coded according to the sequence of processes in the workstation. The weighted time value of the process shall not be higher than the scheduled production rhythm in the workstation. Zero insertion [24] is used to represent the start (or end) position of the activity element of the current workstation. There are n activities and $m - 1$ zeros. The activity element between two adjacent zeros is the same workstation.

3.2 Initial population generation

The quality of the initial population has an obvious impact on the evolution process and the efficiency of the algorithm. In order to ensure the diversity of individuals and solutions in the initial population, the topological sorting random search is used to generate the initial population. The process is as follows.

- According to the diagram of activity priority order, the task without pre-task is found in the complete set of tasks (population) and put into the new set , and the task and its related sequences are deleted in the operation sequence diagram.
- Repeat the above operations until the task set the empty set, and finish the task assignment. The selected task is put into the corresponding gene position in each step, and the obtained sequence is the initial feasible disassembly sequence.

3.3 Decoding

The coding adopts a one-dimension group solution sequence based on the task, which cannot determine the individual's merits and demerits. The solution sequence needs to be allocated to each workstation [25, 26]. The operation is as follows.

- Start the first workstation $j = 1$.
- Initialize the current workstation time and remaining time.
- Find task i in the solution sequence. When the task time allocated exceeds the current remaining time, the assignment fails. Open a new workstation randomly and initialize the current workstation time and remaining time. Else assign task i to the current workstation, update the workstation time and remaining time, and cycle until permutation sequence.

By decoding each individual and inserting zeros between workstations, each workstation and task assignment in the population can be determined, and the visualization of the algorithm can be improved.

3.4 Fitness

Fitness function plays an important role in the evaluation of individuals in GA search evolution. Only the objective function can be used to optimize the system in the solution space. In the Genetic Algorithm space, the objective function is transformed into individual fitness according to certain rules, and the fitness value is evaluated to realize the judgment of the feasible solution in the solution space.

3.5 Genetic operators

Selection

Roulette is the most commonly used selection method. The sampling idea is that the probability of the selected individual inheriting the next generation is directly proportional to the fitness value. The higher the evaluation of the fitness function of the individual, the greater the probability of inheriting to the next generation. This is the probability of an individual selected in function (13).

$$\text{(13)}$$

Crossover

Crossover is an important way to form new individuals. Two chromosomes are selected from the selected population, and some genes are exchanged with specific rules to form new individuals after recombination [27].

Because the random crossover method in traditional methods often leads to a large number of repetitions and conflicts, results in infeasible solutions and affects the operational efficiency of the algorithm, this process uses two-point mapping crossover method, randomly determines two crossover points on the parent chromosome, sorts some genes between the two chromatids of the father generation and adopts partial mapping to save the genes on both sides of the crossover point and put them into a new individual, thus producing two new offspring individuals. Assuming that the third and fifth gene locus are randomly selected as the crossover points, the sequence {6, 3, 7}, {2, 4, 9} before the two crossover points in the parent generation can be preserved, and the sequence {8, 5, 1} between the two crossover points in the parent 2 is {1, 5, 8}. The specific process is shown in Fig. 1.

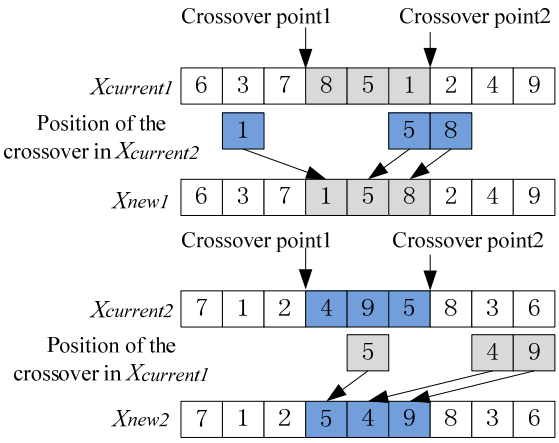


Fig. 1 The diagram of crossover

Mutation

Similar to the crossover operation, the mutation will produce infeasible solutions due to the constraints of the activity priority relationship. In this process, single-point variation is used. A mutation point (position 4 in Fig. 2) is randomly generated on the chromosome of the parent generation. According to the activity priority diagram, the pre-process and post-process of the mutation point are found out, so that the pre-process and its previous gene position, post-process and its subsequent gene position can be retained. Position 4 is randomly inserted into the nearest gene position between the pre-process and the post-process in the chromosome, and then the above genes are integrated to generate new progeny chromosomes. If the selected mutation position does not have an optional insertion position, the mutation point can be re-selected.

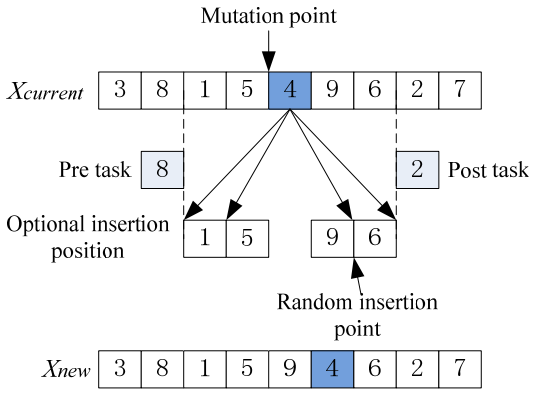


Fig. 2 The diagram of mutation

3.6 Termination conditions

As a search tool of repeated iterations, GA approximates the optimal value infinitely through multiple evolutionary cycles, instead of just calculating the optimal solution. Therefore, it is necessary to determine the termination condition and the generations of genetic iteration. At the beginning of the algorithm, the number of iterations should be set as small as possible, and then increases iterations as appropriate. When the number of iterations exceeds the required maximum number of generations the algorithm stops.

3.7 Implementation process

Although the individuals of the initial population are feasible, it is impossible to determine whether the optimal individuals appear in the early stage of the algorithm. Therefore, the probability of crossover and mutation can be increased to enhance the optimization ability of the algorithm. In the later stage of iteration, when the algorithm converges gradually, the individual fitness value becomes higher. In order not to destroy the excellence of genes in individuals, the probability of crossover and mutation is often reduced to improve the operation efficiency of the algorithm. In this paper, the adaptive crossover and mutation probability are used. The symbols used to run the algorithm are as follows: ὐ is the maximum number of iterations, ά is the number of iterations, ὖ῏ is the crossover probability. ὖῌ is the mutation probability. ὖ῏ῳῳῳῳ is the minimum crossover probability, ὖ῏ῳῳῳῳ is the maximum probability. ὖῌῳῳῳῳ is the minimum mutation probability. ὖῌῳῳῳῳ is the maximum mutation probability.

$$ὖ῏ = ὖ῏ῳῳῳῳ + \frac{ὖ῏ῳῳῳῳ - ὖ῏ῳῳῳῳ}{ὐ} × ά \quad (14)$$

$$ὖῌ = ὖῌῳῳῳῳ + \frac{ὖῌῳῳῳῳ - ὖῌῳῳῳῳ}{ὐ} × ά \quad (15)$$

Step1: Determination of parameters. Select the value of ὔὖ, ὖ῏, ὖῌ.
Step2: Initialization population. Order ὸ = 0, the initial population P 0 with ὔὖ individuals are generated under the condition of the beat constraint and the priority of tasks.
Step3: Fitness assessment. The fitness value of each individual in the ″Y generation population is calculated.
Step4: Selection. ὔ ὖ individuals are selected from the ″Y generation population and cloned into the ″Y + 1 generation.
Step5: Crossover.
Step6: Mutation.
Step7: Optimal preservation strategy.
Step8: Repeat. Order $t \leftarrow$ ὸ + 1, when the termination condition is met, it ends. Else, turn to Step3.

4. Case studies: The practical application and analysis

4.1 Disassembly of the computer components

The disassembly information of a computer component with 8 parts is shown in Table 1. The disassembly of the parts is shown in Fig. 3. The improved Genetic Algorithm is used to solve the problem.

Matlab R2012b software is used to realize the algorithm program on the windows10 system platform, and the above examples are solved [28]. The minimum number of workstations is 4, the hazard index ″O is 7, the demand is ″O = 211, the direction index is ′Y = 7. The optimal solution is given in Table 2. Table 3 shows the optimal disassembly series solution and the workstation after balancing. Among them, disassembly tasks 1 and 5 are assigned to workstation 1, and workstation 2 is mainly responsible for disassembly tasks 3, 6, 2, and so on. In addition, the optimal solution removes high demand parts 3, 6, 2, and hazardous parts 7 earlier, allowing seven directions to change, and the calculation time of the algorithm is less than 1s. The equilibrium and hazard objectives of the solution obtained are the same as those in reference [29], while the demand index is better than that in reference [29], and there is one more direction index than reference [29]. Therefore, the overall performance of the solution obtained is better than that in reference [29].

The optimal solutions obtained by these two single objective algorithms are shown in Table 2, and the parameter design in reference [29] is combined with the DLBP problem. After repeated tests on the quality of the solution and the efficiency of the algorithm, the parameters in this paper are set as follows: ὔὖ = 1, ὐ = 100, ὖ῏ῳῳῳῳ = 0.5, ὖ῏ῳῳῳῳ = 0.95, ὖῌῳῳῳῳ = 0.005, ὖῌῳῳῳῳ = 0.01. After calculation, the optimal value is obtained and the operation time is analyzed.

The optimal disassembly series solutions are shown in Table 3. Comparing the improved Genetic Algorithm with the traditional Genetic Algorithm, it can be seen that the test results are more prominent than the traditional algorithm in dealing with this problem, and the running time is shorter.

Table 1 The disassembly information for eight components

Disassembly tasks	Disassembly operation	Time	Demands	Harmful	Direction
1	Disassembly PC top cover (TC)	14	360	N	-x
2	Disassembly floppy disk driver (FD)	10	500	N	+x
3	Disassembly hard disk driver (HD)	12	620	N	-x
4	Disassembly bottom plate (BP)	18	480	N	-x
5	Disassembly PCI card	23	540	N	+y
6	Disassembly RAM modular	16	750	N	+z
7	Disassembly power supply (PU)	20	295	Y	-x
8	Disassembly mother board (MB)	14	360	N	-x

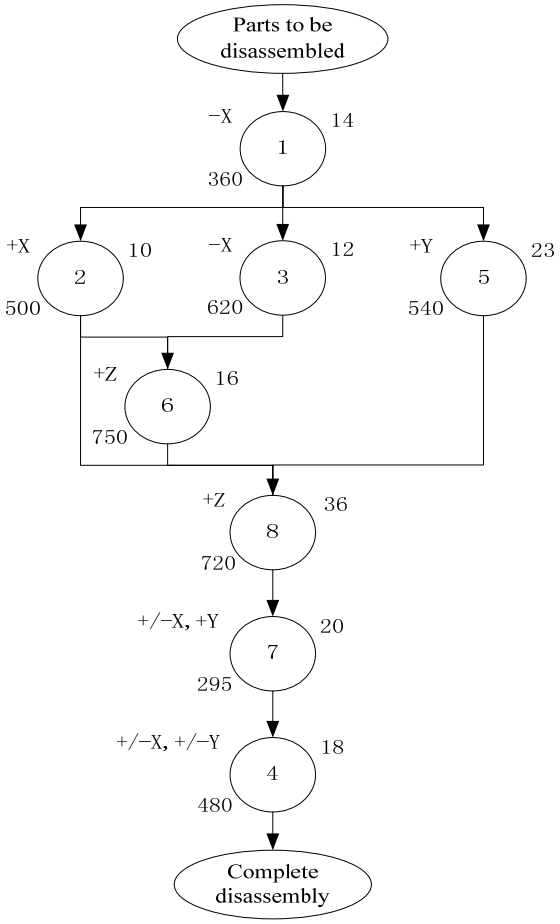


Fig. 3 The activity priority relationship of components

Table 2 The comparison of optimization results

Algorithm	Number of workstations S	Balance index F	Demand quantity D	Harmful index H	Direction index R
Reference [29] GA	4	211	19275	7	-
Improved GA	4	211	19025	7	7

Table 3 The optimal disassembly series solution of DLBP

		Workstation *i*				
		S1	S2	S3	S4	
Disassembly series	1	14				Free time 0 s
	5	23				
	3		12			
	6		16			
	2		10			
	8			36		
	7				20	
	4				18	
Total time		37	38	36	38	
Idle time		3	2	4	2	

4.2 Disassembly of the automobile engine

Taking the automobile engine disassembly example in reference [30] as the research object, the engine is completely disassembled, and Fig. 4 shows the engine disassembly diagram. The original enterprise did not consider the damage and demand of disassembly, and the variability is poor, so it cannot adapt to the change of disassembly task in time. Now the improved Genetic Algorithm is used to improve the engine cylinder block disassembly line. The relevant data are shown in Table 4.

The bearing, toothed belt, belt, connection key, connection pin, and other parts in the assembly are simply removed and replaced with a group of assembly and fasteners to reduce the number of disassembly parts. In addition, the disassembly operation was investigated to obtain detailed disassembly data such as operation time, disassembly priority relationship, constraints, disassembly demands, hazards, and direction changes of each part. Standard operating time (SST) on line disassembly was measured. Standard operating time is the time taken by a skilled worker to complete a process at a normal speed under normal operation. The disassembly operation time is measured by the stopwatch method in industrial engineering, and the optimization goal is taken into account the damage of parts, demand, and the change of operation direction.

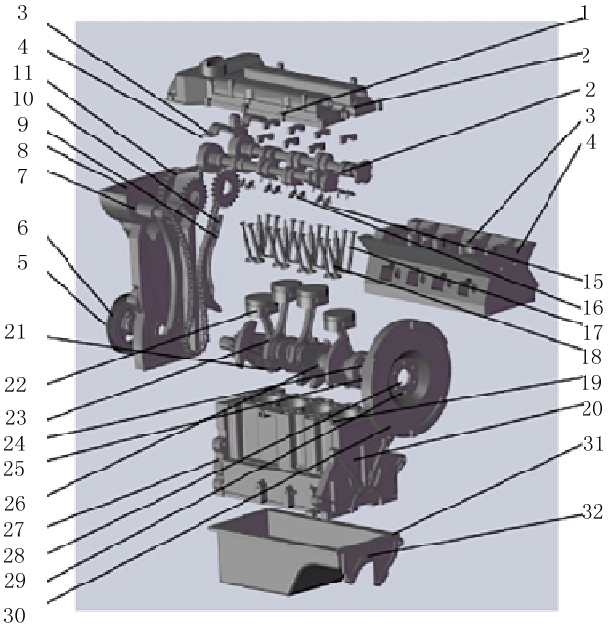


Fig. 4 The engine disassembly diagram

Table 4 The engine part elements

No.	Name	Time	Demand	Harmful	Direction
1	Camshaft coverscrew	5×12	5	N	+z
2	Camshaft cover	3	65	N	+z
3	Camshaft cover plate screw	4×20	15	N	+z
4	Camshaft cover plate	2×8	15	N	+z
5	Starting claw	6.5	35	N	+x
6	Pulley	20	40	N	+x
7	Side cylinder head screw	6.5	45	N	+x
8	Side cylinder head	5.5	65	N	+x
9	Chain restraint screw	4.5×4	8	N	+x
10	Chain restraint mechanism	3	8	N	+x
11	Chain	10	15	N	+z
12	Camshaft	3.2	60	N	+z
13	Cylinder head cover screw	4×8	70	N	+z
14	Top cylinder head cover	21	30	N	+z
15	Rocker fastening screw	8×16	45	N	+z
16	Rocker	3×16	95	N	+z
17	Valve	4×16	50	N	+z
18	spark plug	28×4	40	Y	+z
19	Cylinder block screw	5.5	40	N	+z
20	Cylinder Block	20	95	N	+z
21	Connecting rod cover	10×4	65	N	+z
22	Connecting rod	5×4	40	N	+z
23	Connecting rod screw	7×4	40	N	+z
24	Crankshaft bearing cover screw	5×10	25	N	+z
25	Crankshaft bearing cover	3×5	15	N	+z
26	Crankshaft	20	95	N	+z
27	Cylinder Black	23	30	N	+z
28	Flywheel nut	15	30	N	-x
29	Flywheel screw	5.5×6	45	N	-x
30	Flywheel	30	50	N	-x
31	Seat screw	5×12	65	N	+z
32	Seat	6	30	N	+z

Input the operation sequence matrix of the automobile engine of the disassembly line and program it on Matlab2012b. The given beat ő῾Ῡ is 120s. The parameters of the algorithm are as follows: ŭ ũ = 50, MaxGen = 150, ῾0ö ũ = 0.9. By optimizing multiple objective functions, the algorithm in this paper achieves the prediction results, jumps out of the local optimal solution to obtain a better solution, and the searchability is significantly improved. The optimization results are shown in Fig. 5.

The zeroing operation in encoding and decoding can isolate each workstation. After the operation, the balancing result of linear layout and U-shaped layout is shown in Fig. 6. The number of workstations in linear layout optimization is 15, while the number of U-shaped layouts is 9. Therefore, the U-shaped layout can minimize the number of workstations and has a certain economy.

The station allocation and operating time range volatility are shown in Table 5 and Table 6. The range volatility is the ratio of idle time to total operating time. All the data fluctuate within the 10 % range. It can be concluded that the assignment and optimization of work elements of both linear and U-shaped workstations in DLBP problem are reasonable, which proves the scientific and effectiveness of the model. The utilization rate of each workstation in the disassembly line under the U-shaped layout is higher than that of the traditional straight line, and the idle waiting time in the workstation is smaller, which shows a significant difference between the two types of disassembly lines. It is proved that the U-shaped layout is better and can achieve the balance effect.

The algorithms can find out a better solution, the number of workstations, balance index, resource index got better optimization, such as the direction indicators have improved, can achieve better solving the performance. The feasibility of the proposed improved genetic algorithm in solving the balance problem of a multi-objective U-shaped disassembly line is verified. The optimization results of straight layout and U-shaped layout are compared, and it is proved that U-shaped layout is more suitable for disassembly line layout.

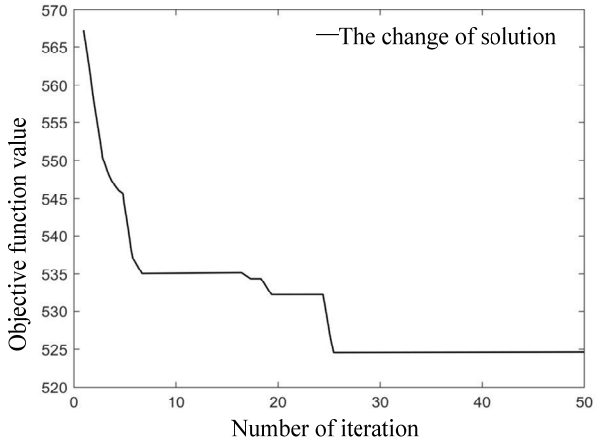
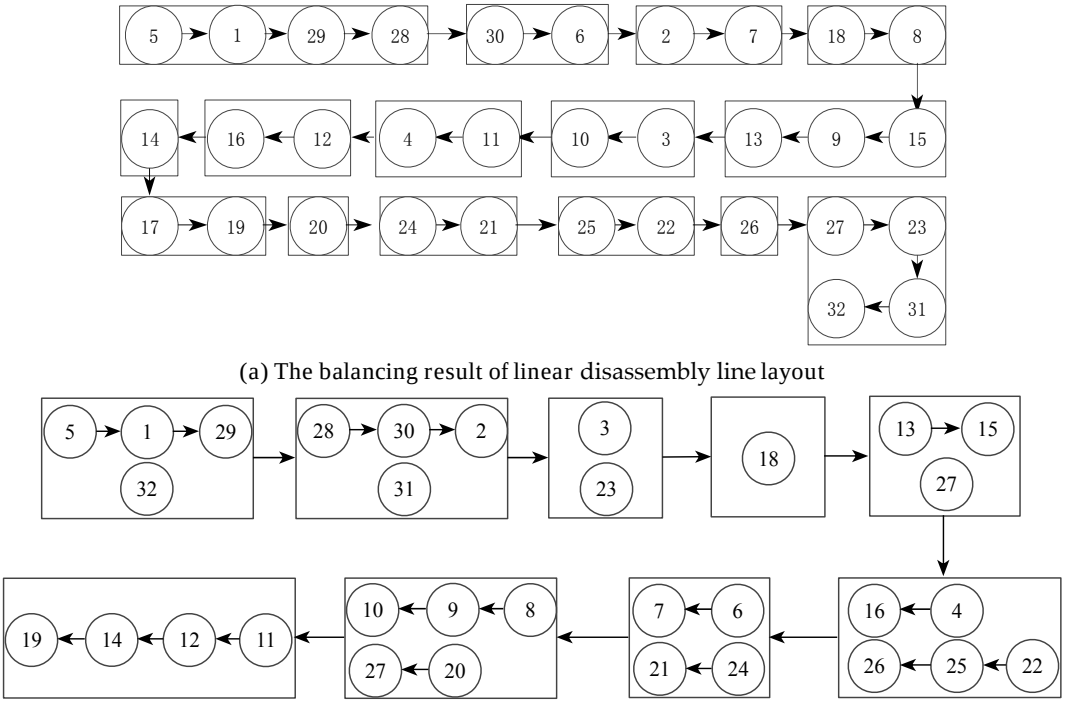


Fig. 5 The optimal results



(a) The balancing result of linear disassembly line layout

(b) The balancing result of U-shped disassembly line layout

Fig. 6 The balancing results

Table 5 The workstation allocation of liner disassembly line layout

Workstation number	Station time (s)	Idle time (s)	Range fluctuation Range volatility (%)
1	114.5	5.5	0.56
2	59.5	60.5	6.13
3	117.5	2.5	0.25
4	72.5	47.5	4.85
5	109	11	1.11
6	114	6	0.61
7	119	1	0.10
8	110	10	1.01
9	55	65	6.58
10	117	3	0.3