Assembly Line Balancing Based on Double Chromosome Genetic Algorithm

Liu Yanhou(刘俨后), *Zuo Dunwen*(左敦稳)*, *Zhang Dan*(张丹)

College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, P. R. China

(Received 4 November 2013; revised 24 January 2014; accepted 25 March 2014)

Abstract: Aiming at assembly line balancing problem, a double chromosome genetic algorithm (DCGA) is proposed to avoid trapping in local optimum, which is a disadvantage of standard genetic algorithm (SGA). In this algorithm, there are two chromosomes of each individual, and the better one, regarded as dominant chromosome, determines the fitness. Dominant chromosome keeps excellent gene segments to speed up the convergence, and recessive chromosome maintains population diversity to get better global search ability to avoid local optimal solution. When the amounts of chromosomes are equal, the population size of DCGA is half that of SGA, which significantly reduces evolutionary time. Finally, the effectiveness is verified by experiments.

Key words: double chromosome; genetic algorithm; assembly line balancing; mathematical model; global optimum

CLC number: TH166; F406 **Document code**: A **Article ID**: 1005-1120(2014)06-0622-07

1 Introduction

Assembly line balancing (ALB) distributes jobs into different stations with some specific constraints, which can rationalize station number and production cycle to balance workload of stations[1]. In 1954, Bryton presented ALB problem and illustrated some solutions[2]. Since then ALB has drawn great attention in the academia. Salveson firstly studied ALB in a way of mathematical analytic and then proposed a linear programming models to solve ALB[3]. Under the condition of fixed cycle time, Jackson minimized the number of stations with enumeration method[4].

In recent years, genetic algorithm (GA) has significantly developed theoretically and practically, and it has been successfully applied to solving ALB. A hybrid GA for ALB was put forward in Ref. [5] and ranked positional weight method was introduced into initialization process. In Ref. [6], the crossover operator, mutation operator and initial population are constructed based on the constrains of assembly jobs, which can ensure the feasibility of all individuals, therefore improve efficiency of the algorithm. According to the features of two-sided assembly[7], a modified GA was proposed to solve ALB, by adopting a combinational encoding scheme based on sequence and task. For a specific problem, GA was combined with heuristic strategy to speed up the algorithm convergence[8]. In Ref. [9] an iterative GA was developed to solve the assembly line worker assignment and balancing problem of type-II. Ref. [10] addressed the operator assignment in predefined workstations of an assembly line, and adopted GA to get sustainable result of fitness function of cycle time, total idle time and output. Ref. [11] proposed a hybrid GA to solve the setup assembly line balancing and scheduling problem. Aiming at the existing mixed-model assembly line balancing problem, Ref. [12] proposed an improved GA to minimize cycle time and workload variance.

The problem of ALB is NP-hard, which will

Foundation items: Supported by the 12th Five-Year Plan National Pre-research Program of China; the Aerospace Science Foundation of China (20111652016); the China Postdoctoral Science Foundation (2012M511748); the Jiangsu Planned Projects for Postdoctoral Research Funds (1102053C).

* **Corresponding author**: Zuo Dunwen, Professor, E-mail: zuodw@nuaa.edu.cn.

result in a combinatorial explosion when the number of jobs increases. Therefore, mathematical analytic methods or usual heuristic methods are not ideal. A double chromosome genetic algorithm (DCGA) is proposed to model the coexistence of dominant gene and recessive gene, which can maintain population diversity during evolution to avoid trapping in local optimum.

2 ALB Problem Description

ALB problems fall into three types[13]: ALBP-1, minimize the station number m when cycle time C is fixed; ALBP-2, minimize cycle time C when station number m is fixed; ALBP-3, minimize smoothness index (SI) when m and C are fixed. The expression of SI as

$$\mathrm{SI}=\sqrt{\sum_{r=1}^{m}(C-T_r)^2} \tag{1}$$

where T_r is the total job time of station r. We take ALBP-2 as the object of our study, and use SI as one of the evaluation items. While C is same, the solution getting smaller SI is better. The mathematical model of ALBP-2 is shown as

$$\min(C.\,\mathrm{SI}) \tag{2}$$

s. t.

$$\sum_{k=1}^{n}x_{kr}=1 \quad \forall r\in \boldsymbol{R} \tag{3}$$

$$\sum_{r=1}^{m'}x_{jr}\leqslant\sum_{r=1}^{m'}x_{ir}\leqslant 1$$

$$\forall a_{ij}=1,\forall m'\in[1,m] \tag{4}$$

$$0\leqslant\sum_{k=1}^{n}(x_{kr}\times t_k) \quad \forall r\in \boldsymbol{R} \tag{5}$$

$$\sum_{k=1}^{n}(x_{kr}\times t_k)\leqslant C \quad \forall r\in \boldsymbol{R} \tag{6}$$

where

n: the number of assembly jobs,

K: the set of jobs, $K=\{k=1, 2, \cdots, n\}$,

R: the set of stations, $R=\{r=1, 2, \cdots, m\}$,

t_k: the time of job k,

x_{kr}: decision variable ($x_{kr}=1$ when job k is allotted in station r),

$\boldsymbol{A}_{n\times n}$: precedence matrix $\boldsymbol{A}_{n\times n}=(a_{ij})$; ($a_{ij}=1$ when job j is the successor of job i).

The function $\min(a.\,b)$ of Eq. (2) means that the variable a has the priority, such as

$\min(a_1.\,b_1)<\min(a_2.\,b_2)$, when $a_1<a_2$, $\forall b_1$, $\forall b_2$.

$\min(a_1.\,b_1)<\min(a_2.\,b_2)$, when $a_1=a_2$, $b_1<b_2$.

Eq. (3) means that each job k has to be allotted in one of the stations once and only once. Eq. (4) satisfies the constraint of $\boldsymbol{A}_{n\times n}$. Eq. (5) means that each station r has to be used. Eq. (6) means that the job time of each station has to satisfy the constraint of C.

3 DCGA

During the evolution of standard genetic algorithm (SGA), one parent individual with higher fitness will have a better chance to reproduce, while one parent individual with lower fitness will be gradually replaced. In the single-chromosome population, some individuals with lower fitness may contain global optimum gene segments which will be disappear because of the lower fitness, while some individuals with higher fitness may merely contain local optimum gene segments, yet these individuals can gradually dominate the evolution which will lead GA to trap in local optimum. Therefore, there is always a defect of GA, which may cause prematurity of evolution[14]. For this characteristic, we propose a kind of genetic algorithm with double chromosome: DCGA.

3.1 Basic flow of DCGA

There are two chromosomes in each individual of DCGA, Ⅰ and Ⅱ, and each of which is one of the values in solution space. And then the better one with higher fitness, as dominant chromosome, determines the fitness of this individual. The basic strategy of DCGA is as follows.

Step 1 Population Initialization: Population P with size N is randomly generated, so the size of chromosomes is $2N$. Then after computing the fitness of chromosome Ⅰ and chromosome Ⅱ of every individual p_i ($i=1, 2, \cdots, N$), the fitness of p_i equals to that of dominant chromosome.

Further, the best individual is singled out as p_0.

Step 2 Selection: Individual p_i is randomly selected on the basis of fitness, and then is copied into population SP. During selection, one individual with higher fitness has a greater chance to be copied multiple times, whereas one individual with lower fitness may be eliminated. In this way, SP with size N is obtained by N selections, and then P is replaced by SP. Now P is the new parent population.

Step 3 Crossover: By traversing through P, p_i is chosen into population CP with probability r_c. The last choice will be aborted if the size of CP is odd number. By randomly offering one of their two chromosomes, cp_i and cp_{i+1} ($i=1, 3, \cdots$) are crossed, as shown in Fig. 1. After traversing through CP, all cp_i is transferred into P. Now P is the new offspring population.

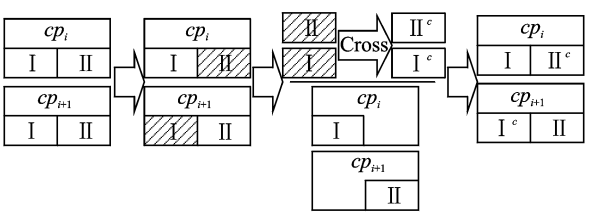

Fig. 1 Crossover of double chromosomes

Step 4 Mutation: By traversing through P, p_i is chosen into population MP with probability r_m, and both chromosomes of mp_i ($i=1, 2, \cdots$) are mutated. After traversing through MP, all mp_i are transferred into P. Now P is the new offspring population.

Step 5 Evaluation: After computing fitness of all p_i, p_0 will be replaced by the best one if its fitness is higher than that of p_0. At this moment, if the terminal condition (such as iterations, rate of evolution, etc.) occurs, end the algorithm, otherwise go to Step 2.

3.2 Performance comparison

To solve multi-objective flexible job-shop scheduling, Ref. [15] introduced an encoding of GA, in which real encoding of process unites with real encoding of process station to generate more outstanding individuals. A new GA, comprised of a rectangle chromosome and a junction chromosome, was proposed by Ref. [16] for solving two-dimensional rectangle packing problem. Unlike the above algorithms with double chromosome structures, DCGA has two identical chromosomes to model the dominant gene and recessive gene. In nature, both of them have the same access to take part in reproduction while produce different effects. In DCGA, dominant chromosome retains good genes to fasten the algorithm convergence, while recessive chromosome maintains population diversity to avoid trapping in local optimum. In the cases of equal chromosomes size, the population size of DCGA is half of SGA′s, and so the decrease of population size can reduce the time requirement of algorithm to speed up the evolution process.

Traveling salesman problem (TSP) is a typical NP-hard combination optimization problem, and the combination scale of n-city TSP is $(n-1)!/2$. In Ref. [17], a 20-city TSP was given, and there were $(20-1)!/2=6.082\,255\times10^{16}$ solutions in its solution space. The coordinates of this 20-city TSP are shown as Table 1 [17].

The optimal solution of 20-city TSP is shown as Fig. 2, and its value is 24.523 8. The density of point distribution in the lower right corner is big, that can lead algorithm trapping in local optimum. This 20-city TSP will be solved by DCGA and SGA respectively with encoding scheme, crossover operator and mutation operator from Ref. [17]. The population size of DCGA is set to 100, while that of SGA is set to 200 to ensure chromosome size equal. The crossover probability is 0.3, and the crossover probability is 0.1. The maximum number of iterations (300 times) is used as the termination condition. The 20-city TSP is repeatedly solved 50 times, and the result is shown in Table 2.

Table 1 Coordinates of TSP (20 cities)

No.	x	y	No.	x	y	No.	x	y
1	5.294	1.558	8	3.447	2.111	15	2.710	3.140
2	4.286	3.622	9	3.718	3.665	16	1.072	3.454
3	4.719	2.774	10	2.649	2.556	17	5.855	6.203
4	4.185	2.230	11	4.399	1.194	18	0.194	1.862
5	0.915	3.821	12	4.660	2.949	19	1.762	2.693
6	4.771	6.041	13	1.232	6.440	20	2.682	6.097
7	1.524	2.871	14	5.036	0.244			

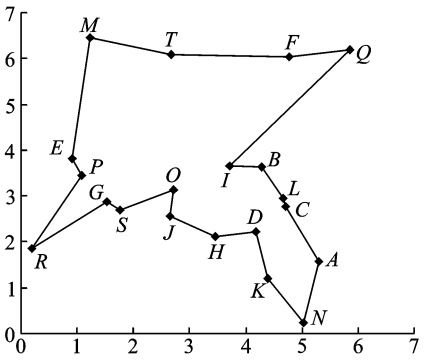

Fig. 2 Optimal solution of 20-citiy TSP

Table 2 Comparison of SGA and DCGA

Testing case	Rate of optimal	Average run-time/s
SGA	10/50	3.7
DCGA	49/50	2.1

Based on TSP, the optimal rate of SGA is 20%, whereas that of DCGA is 98%. Therefore, DCGA has a significant advantage in searching global optimum, and the evolution speed of DCGA is faster than SGA.

4 Solution of ALBP-2 by DCGA

We look into ALBP-2, which minimizes C when m is fixed, and consider SI as one evaluation item. The mathematical model is shown in Eqs. (2－6). Aiming at ALBP-2, DCGA is designed as follows.

4.1 Encoding and decoding

Adopting job-sequence encoding format, all jobs are encoded in the order of distribution, and each gene-bit of chromosome represents a job. As Fig. 3 shows, job 1 is the first, job 2 is the second, …, job 4 is the ninth and job 3 is the last.

Assembly order	1	2	3	4	5	6	7	8	9	10
Job index	1	2	5	10	7	9	6	8	4	3

Fig. 3 Chromosome encoding of 10 jobs

The goal of decoding is to get distribution scheme according to minimum C, which is unknown in ALBP-2. For this matter, an estimated cycle time C^* is given, which will be increased progressively to search a suitable C in decoding process. In the actual search process, incremental step length is not continuous but discrete based on single job time. The decoding process is shown as:

Step 1 Compute the theoretical minimum cycle time $C^t=(\Sigma t_k)/m$, where Σt_k is the sum of all jobs time, and m the number of stations. Set $C^*=C^t$.

Step 2 On the basic of C^* as cycle time, n jobs are distributed in m stations according to the coding sequence, and then station time are T_1, T_2, …, T_r, …, T_m. If each station time $T_r \leqslant C^*$ $(r=1, 2, \cdots, m)$, the C^* will be the practical minimum cycle time in this case of coding sequence $(C=C^*)$, then end it or else go to Step 3.

Step 3 Compute every potential increment $d_r(r=1, 2, \cdots, m)$ of all stations, $d_r(r=1, 2, \cdots, m-1)$ is equal to the first job time of station $r+1(r=1, 2, \cdots, m-1)$, and set $d_m=-d_{m-1}$, as shown in Fig. 4.

Step 4 Set $C=\max\{T_r\}$, $C^*=\min\{T_r+d_r\}$ $(r=1, 2, \cdots, m)$, as shown in Fig. 4. If $C \leqslant C^*$, C will be the practical minimum cycle time in this case of coding sequence, and then end it or else go to Step 2.

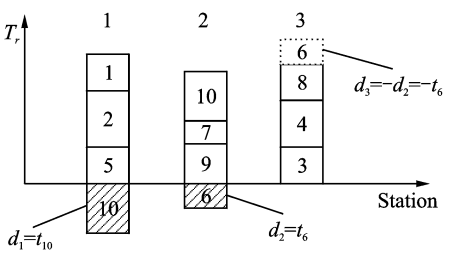

Fig. 4 Potential increment d_r

4.2 Fitness evaluation

As a standard to evaluate individuals, the evaluation function of fitness is the only basis of survival of the fittest. The minimum C is taken as the optimization objective, and the SI is considered as secondary evaluation item, as shown in Eq. (2).

$$F_1=\min(C),\ C=\max(T_r) \tag{7}$$

$$F_2=\min(\mathrm{SI}),\ \mathrm{SI}=\sqrt{\sum_{r=1}^{m}(C-T_r)^2} \tag{8}$$

The primary evaluation item C, which is set $C=\max\{T_r\}$ $(r=1, 2, \cdots, m)$, based on distribution scheme, is shown in Eq. (7). If C values of the solutions are equal, the values of SI will be evaluated, as shown in Eq. (8).

4.3 Crossover operator

One of the two chromosomes, Ⅰ and Ⅱ, is randomly selected to participate in crossover process, as shown in Fig. 1. Paired-point crossover operator is adopted to find two random points, as shown in Fig. 5 (Ⅰ of parent-1 and Ⅱ of parent-2 are selected), and the gene segments between these two points of parent-1 are replaced by new arrangement which appears evolving from parent-2 to offspring-1. Likewise, offspring-2 evolves from parent-2. The offsprings are shown in Fig. 6. This kind of crossover operator keeps the chromosomes of offspring feasible, which can reduce calculation amount while increase efficiency.

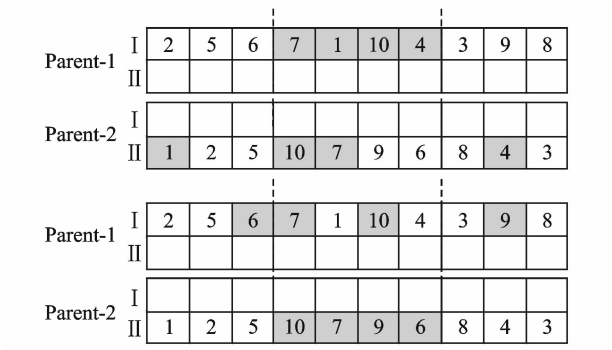

Fig. 5 Paired-point cross operator

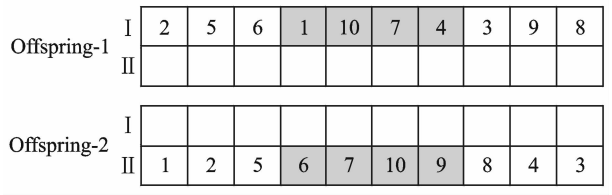

Fig. 6 Encoding of offspring

4.4 Mutation operator

Through randomly finding 2 points, both of the two chromosomes, Ⅰ and Ⅱ, mutate independently. The mutation operator takes approach that rearranging one of these three gene segments. The rearrangement is a allocation process from that random point according to the precedence matrix. The parent and its offspring are shown in Fig. 7.

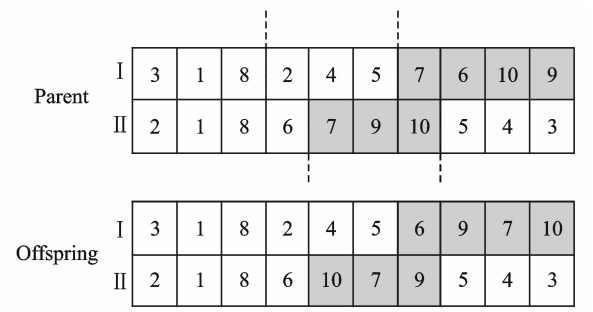

Fig. 7 Mutation operator of chromosome

5 Case Study

The Buxey problem is a testing case of ALB problem[6,18]. The directed graph of Buxey's jobs priority is shown as Fig. 8, and the job time is listed in Table 3.

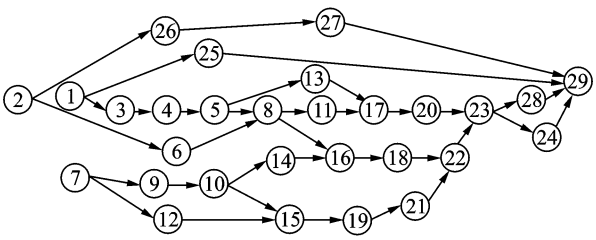

Fig. 8 Precedence relation of Buxey-problem

Table 3 Job time of Buxey-problem

No.	t_k	No.	t_k	No.	t_k	No.	t_k	No.	t_k
1	7	7	8	13	9	19	10	25	14
2	19	8	16	14	4	20	16	26	2
3	15	9	2	15	14	21	1	27	10
4	5	10	6	16	7	22	9	28	7
5	12	11	21	17	14	23	25	29	20
6	10	12	10	18	17	24	14		

The population size of DCGA is set to 100. The crossover probability is 0.6, and the crossover probability is 0.4. Taking iterative 100 0 times as the terminal condition, the solution on Buxey problem is shown in Table 4.

DCGA and FTSGA[6] can find the minimums of C of all different m, and GGA[18] fails to get minimum C when $m=13$. Although the minimums of C of DCGA and FTSGA are equal, SI of DCGA is a smaller SI (or SI/m). As shown in Table 4, DCGA solution is much better than that of GGA and slightly better than that of FTSGA

in the solution of Eq. (2).

On other different scaled problems, Scholl problem set from http://alb.mansci.de/ is used, and Jackson, Kilbrid, Wee-Mag, Mukherje, Barthol2 and Scholl are solved by DCGA. Solutions on Scholl problem set are shown in Table 5.

On Jackson, Kilbrid, Wee-Mag, Mukherje ($m=10,18$), Barthol2 and Scholl ($m=30$, 50) problems, as shown in Table 5, DCGA can get all optimal solutions (minimum C) and get a smaller SI. On Mukherje ($m=25$) and Scholl ($m=40$), DCGA gets satisfying quasi optimum values. Tests indicate that DCGA is competent to solve different scaled problems.

Table 4 Solution on Buxey problem

m	T_r/s	DCGA		GA based on feasible task sequence[6]		Grouping GA[18]	
		C	SI/m	C	SI/m	C	SI/m
7	47,47,47,47,44,46,46	47	0.474	47	0.474	47	0.474
8	40,41,41,40,40,40,41,41	41	0.250	41	0.250	41	0.500
9	37,37,37,37,37,36,36,33,34	37	0.577	37	0.657	37	0.657
10	31,32,32,32,33,32,33,33,34,32	34	0.566	34	0.600	34	0.849
11	29,29,30,29,28,30,30,28,32,29,30	32	0.833	32	0.833	32	1.233
12	27,28,27,28,28,27,27,26,26,25,28,27	28	0.391	28	0.408	28	0.456
13	26,25,23,26,25,23,24,24,25,27,25,24,27	27	0.675	27	0.675	30	—
14	22,24,22,23,22,24,24,23,25,24,24,23,24,20	25	0.598	25	0.631	25	0.833

Table 5 Solutions on Scholl problem set

Problem	n	Sum of t_k	m	Min C	Solution C	Solution SI/m
JACKSON	11	46	3	16	16	0.141
			4	12	12	0.141
			5	10	10	0.245
KILBRID	45	552	8	69	69	0.370
			9	62	62	0.483
			10	56	56	0.346
WEE-MAG	75	1 499	10	150	150	0.522
			20	77	77	0.681
			30	56	56	1.257
MUKHERJE	94	4 208	10	424	424	0.903
			18	239	239	1.145
			25	172	175	1.973
BARTHOL2	148	4 234	30	142	142	0.327
			40	106	106	0.520
			50	85	85	0.810
SCHOLL	297	69 655	30	2 322	2 322	1.016
			40	1 742	1 747	2.147
			50	1 394	1 394	1.243

6 Conclusions

As a typical combination optimization problem, ALB problem is extremely complex. Aiming at ALBP-2, GCDA with two identical chromosomes in each individual is proposed. In DCGA, the dominant chromosome reserves excellent gene segments to fasten convergence but the recessive chromosome maintains population diversity to avoid trapping in local optimum. In equal chromosomes size environment, the population size of DCGA is half of that of SGA which can speed up evolution by saving costs of algorithm running time. By Scholl problem set, the effectiveness of DCGA is verified to provide a new solution for ALB problem.

References:

[1] Cao Zhenxin, Zhu Yunlong, Li Fuming. Research on dynamic planning & simulation optimization of mixed model general automobile assembly lines[J]. Computer Integrated Manufacturing Systems, 2006, 12(4):526-532. (in Chinese)

[2] Driscoll J, Thilakawardana D. The definition of assembly line balancing difficulty and evaluation of balance solution quality[J]. Robotics and Computer Integrated Manufacturing, 2001,17(1/2):81-86.

[3] Salveson M E. The assembly line balancing problem [J]. Journal of Industrial Engineering, 1955,6(3): 18-25.

[4] Jachson J R. A computing procedure for a line balancing problem[J]. Management Science, 1956,2(3): 267-271.

[5] Noorul Haq A, Rengarajan K, Jayaprakash J. A hybrid genetic algorithm approach to mixed-model assembly line balancing[J]. International Journal of

Advanced Manufacturing Technology, 2006, 28(3/4):337-341.

[6] Pi Xingzhong, Fan Xiumin, Yan Junqi. Applying the genetic algorithm based on feasible task sequence to ALB_2[J]. Journal of Shanghai Jiaotong University, 2005,39(7):1123-1127. (in Chinese)

[7] Wu Erfei, Jin Ye, Xu Aimin, et al. Two-sided assembly line balancing based on modified genetic algorithm[J]. Computer Integrated Manufacturing Systems, 2007,13(2):268-274. (in Chinese)

[8] Kim Yong Ju, Kim Yeo Keun, Cho Yongkyun. A heuristic-based genetic algorithm for workload smoothing in assembly lines[J]. Computers and Operations Research, 1998,25(2):99-111.

[9] Mutlu O, Polat O, Supciller A A. An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II[J]. Computers & Operations Research, 2013,40(1):418-426.

[10] Zaman T, Paul S K, Azeem A. Sustainable operator assignment in an assembly line using genetic algorithm[J]. International Journal of Production Research, 2012,50(18):5077-5084.

[11] Yolmeh A, Kianfar F. An efficient hybrid genetic algorithm to solve assembly line balancing problem with sequence-dependent setup times[J]. Computers and Industrial Engineering, 2012,62(4):936-945.

[12] Tang Q, Liang Y. The improved genetic algorithm for balancing mixed-model assembly line[C]//2011 International Academic Conference on Numbers, Intelligence, Manufacturing Technology and Machinery Automation, MAMT 2011. Wuhan, China: Trans Tech Publications, 2012:603-608.

[13] Waldemar G. Cycle time in assembly line balancing problem[C]//21st International Conference on Systems Engineering. Las Vegas: Proc.-ICSEng, 2011: 171-174.

[14] Liu Ran, Lou Peihuang, Tang Dunbing, et al. A vaccine-symbiosis clonal selection algorithm for mixed-model scheduling on assembly lines[J]. Journal of South China University of Technology: Natural Science Edition, 2010, 38(3): 133-137. (in Chinese)

[15] Zhu Wenlong, Ding Huafu. Application of genetic algorithms in multi-objective flexible job-shop scheduling[J]. Computer Technology and Development. 2009,19(4):217-219.

[16] Matayoshi Mitsukuni. Double chromosome GA with corner junction for solving the 2D strip packing problem [C]//36th Annual Conference of the IEEE Industrial Electronics Society. Glendale: IECON Proc, 2010:1110-1116.

[17] Wang Xiaoping, Cao Liming. T genetic algorithm: Theory, application and software implementation [M]. Xi′an: Xi′an Jiaotong University Press, 2002: 123-129. (in Chinese)

[18] Rekiek B, De Lit P, Pellichero F, et al. Applying the equal piles problem to balance assembly lines [C]//International Symposium on Assembly and Task Planning. Porto Portugal: [s. n.], 1999:399-404.

(Executive editor: Zhang Bei)