doi:10.3772/j.issn.1006-6748.2016.02.001

Buffer allocation method of serial production lines based on improved ant colony optimization algorithm①

Zhou Binghai (周炳海)②, Yu Jiadi
(School of Mechanical Engineering, Tongji University, Shanghai 201804, P. R. China)

Abstract

Buffer influences the performance of production lines greatly. To solve the buffer allocation problem (BAP) in serial production lines with unreliable machines effectively, an optimization method is proposed based on an improved ant colony optimization (IACO) algorithm. Firstly, a problem domain describing buffer allocation is structured. Then a mathematical programming model is established with an objective of maximizing throughput rate of the production line. On the basis of the descriptions mentioned above, combining with a two-opt strategy and an acceptance probability rule, an IACO algorithm is built to solve the BAP. Finally, the simulation experiments are designed to evaluate the proposed algorithm. The results indicate that the IACO algorithm is valid and practical.

Key words: buffer allocation, improved ant colony optimization (IACO) algorithm, serial production line, throughput rate

0 Introduction

The buffer allocation problem (BAP) is a significant optimization problem faced by engineers of manufacturing system, which refers to the way of allocating buffer storage within the production line. While buffers can compensate for the blocking and starving of stations in the production line, inclusion of buffers results in additional costs probably due to increased capital investment, floor space and in-process inventory. Therefore determining appropriate buffer storage sizes is still a challenging problem.

Due to its importance and complexity, several authors have been working on the BAP for many years. Ref. [1] developed a simulated annealing approach for solving BAP in reliable production lines with the objective of maximizing their average throughput. Ref. [2] presented five different search algorithms to solve the BAP of reliable production lines, including the genetic algorithm (GA), tabu search, simulated annealing, myopic and complete enumeration. Ref. [3] proposed an artificial neural network and myopic algorithm based decision support system on reliable production lines. However, the aforementioned literature only focused on reliable production lines. Ref. [4] proposed a quantitative method to determine the buffer size in front of the bottleneck under multi-product. Ref. [5] developed a new efficient simulation model and an experimental cross matrix for serial production lines to determine the optimal buffer size. Ref. [6] and Ref. [7] proposed an exact Markovian model and an approximate analytical method for unreliable serial flow lines to analyze the relationship between throughput and buffer capacity, respectively. But the aforementioned literature only studied an unreliable serial flow line with two workstations and an intermediate buffer. Ref. [8] implemented a combined artificial immune system optimization algorithm in conjunction with a decomposition method to allocate buffers in transfer lines for maximizing economic profit and throughput. Ref. [9] presented a GA and simulation to solve the BAP of flexible manufacturing system. However, the drawback of these meta-heuristics such as GA in solving combinatorial optimization problems is the necessity to set a number of uncertain parameters, which significantly increases the search time and the number of evaluated solutions to find the optimal or near optimal solution. Ref. [10] developed a petri-net based simulation model to study the continuous flow transfer line with three machines and two buffers, and then analyzed the relationship between the equipment reliability and buffer capacity. Ref. [11] presented a local search based degraded ceiling (DC) approach for solving the BAP. However, the objective function may not be a monotone increasing function as the search time goes.

① Supported by the National Natural Science Foundation of China (No. 61273035, 71471135).
② To whom correspondence should be addressed. E-mail: bhzhou@tongji.edu.cn
Received on Feb. 12, 2015

In this paper, an improved ant colony optimization (IACO) algorithm is used to solve the BAP. Recently, the ant colony optimization (ACO) algorithm has been successfully used by many scholars to solve combinatorial optimization problems[12,13]. It has many good features: distribution, positive feedback, and robustness[14]. However, ACO may lead to a local optimal solution. Thus, some corresponding improvements are done to prevent it from local optimization. One is that when the algorithm is stagnated, the pheromone intensity is reset on all paths in order to break out of the stagnation. Secondly, when the near optimal solution is found, some changes will be made by two-opt strategy to get new solutions. Thirdly, an acceptance probability rule of simulated annealing for updating the best solution is combined with the algorithm. Simulation results indicate that the proposed approach can lead to results that are consistent with our expectations.

1 Problem description

In this paper, the BAP in a serial production line with unreliable machines is examined, as depicted in Fig. 1, where the rectangles represent machines $M_i(i = 1,\cdots,k)$ and the circles indicate buffers $B_i(i = 1,\cdots,k-1)$. The assumptions of the BAP in a serial production line are listed as follows: 1) Parts go through each of the machines and buffers in sequence, from machine M_1 to M_k. 2) The processing times of all parts are constant and equal for all machines, and the transportation time is negligible. 3) Machines are subject to breakdowns. Times to failure and times to repair for machines are exponentially distributed. 4) Machine M_i is starved at time t if M_{i-1} is down and buffer B_{i-1} is empty; Machine M_i is blocked at time t if M_{i+1} is down and buffer B_i is full. 5) The first machine is never starved, and the last machine is never blocked.

Fig. 1 Serial production line

To solve the BAP, evaluation and optimization tools are needed. The evaluation tool is used to calculate performance measures of production lines which have to be optimized (e.g., the average throughput). A Dallery-David-Xie (DDX) algorithm is applied which is proposed in Ref. [15] to calculate the throughput rate of all new configurations.

As shown in Fig. 2, the principle of DDX algorithm is to decompose a k-machine line L into a set of $k-1$ two-machine lines. Each line $L(i)$ is composed of an upstream machine $M_u(i)$ and a downstream machine $M_d(i)$, separated by a buffer B_i. The procedural form of this method is given as follows:

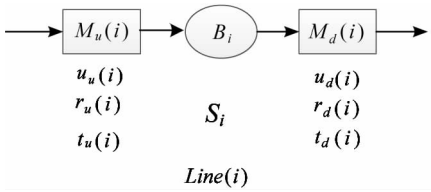

Fig. 2 Decomposition method

Step 1: Initialization:

$r_u(1) = r_1$, $u_u(1) = u_1$

$r_d(i) = r_{i+1}$, $u_d(i) = u_{i+1}(i = 1,2,\cdots,k)$

where $r_u(i)$ and $u_u(i)$ denote the failure rate and repair rate of the upstream machine, respectively; $r_d(i)$ and $u_d(i)$ denote the failure rate and repair rate of the downstream machine, respectively; r_i is the failure rate of machine M_i, $r_i = 1/MTBF_i$; u_i is the repair rate of machine M_i, $u_i = 1/MTTR_i$; $MTBF$ and $MTTR$ represent the mean time between failures and the mean time to repair, respectively.

Step 2: For any $i = 2,3,\cdots,k-1$:

$$I_u(i) = \frac{1}{E(i-1)} + \frac{1}{e_i} - I_d(i-1) - 2 \quad (1)$$

$$u_u(i) = x \times u_u(i-1) + (1-x) \times u_i \quad (2)$$

$$r_u(i) = I_u(i) \times u_u(i) \quad (3)$$

$$e_i = \frac{u_i}{r_i + u_i} \text{ and } x = \frac{P_s(i-1)}{I_u(i) \times E(i)} \quad (4)$$

$$P_s(i) = 1 - \frac{E(i)}{e_d(i)} \quad (5)$$

where $I_u(i)$ and $I_d(i)$ are the ratio of $r_u(i)$ to $u_u(i)$ and $r_d(i)$ to $u_d(i)$, respectively; $E(i)$ is the efficiency of line $L(i)$; e_i is the isolated efficiency of machine M_i; $P_s(i)$ denotes the probability of downstream machine being starved; $e_d(i)$ is the isolated efficiency of downstream machine $M_d(i)$.

Step 3: For any $i = k-2, k-1\cdots2,1$:

$$I_d(i) = \frac{1}{E(i+1)} + \frac{1}{e_{i+1}} - I_u(i+1) - 2 \quad (6)$$

$$u_d(i) = y \times u_d(i+1) + (1-y) \times u_{i+1} \quad (7)$$

$$r_d(i) = I_d(i) \times u_d(i) \quad (8)$$

$$e_i = \frac{u_i}{r_i + u_i} \text{ and } y = \frac{P_b(i+1)}{I_d(i) \times E(i+1)} \quad (9)$$

$$P_b(i) = 1 - \frac{E(i)}{e_u(i)} \quad (10)$$

where $e_u(i)$ is the isolated efficiency of upstream machine $M_u(i)$; $P_b(i)$ denotes the probability of upstream machine being blocked.

Step 4:

If $I_u(i) \neq I_d(i)$:

$$E(i) = \frac{I_d(i) \times e^{\alpha \times S_i} - I_u(i)}{I_d(i) \times (1 + I_d(i)) \times e^{\alpha \times S_i} - I_u(i) \times (1 + I_u(i))} \tag{11}$$

$$\alpha = \frac{2}{t_u(i) + t_d(i)}(\lambda_d(i)u_u(i) - \lambda_u(i)u_d(i)) \times \left(\frac{1}{\lambda_u(i) + \lambda_d(i)} + \frac{1}{u_u(i) + u_d(i)}\right) \tag{12}$$

where $t_u(i)$ and $t_d(i)$ denote the processing time of machine $M_u(i)$ and $M_d(i)$; S_i represents the capacity of the ith buffer.

If $I_u(i) = I_d(i) = I$:

$$E(i) = \frac{\frac{1 + I}{I}S_i + \frac{r_u(i) + r_d(i)}{r_u(i) \times r_d(i)} \times \frac{1}{t_u(i)}}{\frac{(1 + I)^2}{I}S_i + (1 + 2I)\frac{r_u(i) + r_d(i)}{r_u(i) \times r_d(i)} \times \frac{1}{t_u(i)}} \tag{13}$$

Step 5: If $E(1) = E(2) = \cdots = E(k-1)$, stop the procedure, otherwise go to step 2.

On the basis of the descriptions mentioned above, the throughput rate of production line is written as follows:

$$E = f(S) = f(S_1, S_2, \cdots, S_{k-1}) \tag{14}$$

Therefore, the mathematical model for the BAP can be formulated as follows:

$$\text{Maximize } E = f(S) = f(S_1, S_2, \cdots, S_{k-1}) \tag{15}$$

Subject to

$$\sum_{i=1}^{k-1} S_i = N \tag{16}$$

$$0 \leqslant S_i \leqslant S_{iup} (i = 1, \cdots, k-1) \tag{17}$$

S_i nonnegative integers $(i = 1, \cdots, k-1)$ (18)

where N is the total buffer capacity, which is a fixed nonnegative integer; $f(S_1, S_2, \cdots, S_{k-1})$ is the throughput rate of the production line to be maximized; S_{iup} is the upper bound for each of the buffer locations.

For a production line with k machines and N total buffer capacity, the number of possible buffer allocation configurations can be calculated as follows, which is presented in Ref. [11]:

$$C_{N+k-2}^{k-2} = \frac{(N+1)(N+2)\cdots(N+k-2)}{(k-2)!} \tag{19}$$

As for the optimization tool, it is a search method that tries to find an optimal or a near optimal solution which in our case is the capacity of each buffer in a production line. Therefore, a new optimization method based on IACO algorithm is proposed in the next section.

2 Proposed algorithm

ACO algorithm is a novel biomimetic algorithm. Scholars have solved some difficult problems in discrete system optimization based on the behavior of ants seeking a path between their colony and a source of food[16]. When ants seek for food, the front ones release pheromones on the paths they have visited, then the following ones will randomly choose one path according to the pheromones. When the cycle repeats, the shorter path will have a stronger pheromone trail more quickly. After a certain period of time, all the ants will choose the short trail. The procedure of standard ACO is shown in Fig. 3.

As mentioned in introduction, standard ACO may lead to a local optimal solution. In the next sections, details of IACO are provided which is improved by combining with a two-opt strategy and an acceptance probability rule.

```
1. Initialization: set parameters and initialize variables
2. While stop criterion is not satisfied do
3.   While the end node has not been visited do
4.     Every ant chooses a path based on pheromones randomly
5.   end while
6.   Update pheromones
7.   Evaluate all solutions and update the best one
8. end while
```

Fig. 3 Pseudocode of standard ACO

2.1 Encoding

For the BAP, the feasible solution can be expressed as $S = \{S_1, S_2, \cdots, S_{k-1}\}$. It is assumed that there is a certain number of F_i paths in front of the ith buffer, where $F_i < S_i$. If the jth path in front of the ith buffer is selected, the initial solution S should be changed correspondingly.

If $1 \leqslant j < (F_i + 1)/2$:

$$S'_i = S_i + j, S'_{i+1} = S_{i+1} - j \tag{20}$$

If $(F_i + 1)/2 \leqslant j \leqslant F_i$:

$$S'_i = S_i - [j - (F_i + 1)/2] \tag{21}$$

$$S'_{i+1} = S_{i+1} + [j - (F_i + 1)/2] \tag{22}$$

where S'_i is the adjusted capacity of the ith buffer.

2.2 Initialization

Set the initial buffer allocation $S_i \leftarrow N/(k-1)$ and any remaining resource is placed in the middle location.

2.3 Searching

Firstly, all the ants are placed in the first buffer. Then the ants choose paths with the probability p_{ij}^n, until the last buffer B_{k-1} is visited. According to the paths which every ant selects, the corresponding buffer allocation configuration $S = \{S_1, S_2, \cdots, S_{k-1}\}$ is obtained. In the selection phase, probability p_{ij}^n of the nth ant se-

lecting the jth path in front of buffer i is:

$$p_{ij}^{n} = \frac{[\tau_{ij}]^{\alpha}}{\sum_{j=1}^{F_i}[\tau_{ij}]^{\alpha}} \quad (23)$$

where τ_{ij} is the pheromone intensity on each path; α is a constant.

2.4 Updating

For all the new buffer allocation solutions, throughput rate E can be calculated and the optimal solutions $S_{\max}$ can be found among them. Then update the pheromone intensity τ_{ij}. The update rule is given as follows:

$\tau_{ij} \leftarrow \rho \times \tau_{ij} + \Delta\tau_{ij}$, where $1 - \rho$ represents the evaporation of pheromone intensity; $\Delta\tau_{ij}$ is the pheromone increment in this cycle.

$\Delta\tau_{ij} = \sum_{n=1}^{m}\Delta\tau_{ij}^{n}$, where $\Delta\tau_{ij}^{n}$ denotes the pheromone intensity on the path of the nth ant; m denotes the number of ants.

$\Delta\tau_{ij}^{n} = \gamma \times (\frac{E_T^n}{E_T^{n*}})^{\beta} \times E_T^n$, where γ and β are constants; E_T^n is the throughput rate obtained by the nth ant in this cycle; E_T^{n*} is the best obtained solution of all the ants in this cycle.

2.5 Two-opt strategy

In order to prevent the algorithm falling into the local optimization, some changes are done in the near optimal solution by the two-opt strategy:

$$S_i \leftarrow S_i - 1 \text{ and } S_j \leftarrow S_j + 1 \quad (24)$$

Two buffer locations i and j($i, j \in \{1,2,\cdots,k-1\}$ and $i \neq j$) are randomly chosen. Then a new buffer solution can be obtained using Eq. (24). For a production line with k machines, the total number of new possible buffer configurations is $(k-1) \times (k-2)$.

2.6 Acceptance probability rule

In order to improve the exploring capability of the algorithm, an acceptance probability rule of simulated annealing for updating the best solution is introduced. After a new solution is generated, firstly the objective values of old solution S and new solution S' should be calculated. If throughput differential $\Delta E = f(S') - f(S)$ is nonnegative, the new solution is accepted as the best solution, otherwise it is accepted with the probability of $\exp(-\Delta E/T)$, where T is a global time-varying parameter called the temperature.

2.7 Procedure of the IACO

The proposed IACO is formally described as follows:

Step 1: initialize the buffer allocation according to 2.2;

Step 2: initialize the pheromone intensity $\tau_{ij} \leftarrow c$ and the pheromone increment $\Delta\tau_{ij} \leftarrow 0$;

Step 3: choose the paths with probability p_{ij}^{n}, till the last buffer B_{k-1} according to 2.3;

Step 4: evaluate the fitness value according to DDX algorithm and obtain the best solution according to 2.6;

Step 5: perform the two-opt strategy according to 2.5;

Step 6: update pheromone intensity τ_{ij} of all the paths according to 2.4;

Step 7: if it has led to a local optimal solution, go to step 2, otherwise go on;

Step 8: if the termination criterion is not met, initialize $\Delta\tau_{ij} \leftarrow 0$, and go to step 3; otherwise, stop and record the optimal buffer allocation solution S^* and its throughput rate E^*.

3 Numerical examples

In order to evaluate the applicability of the IACO algorithm, experiments are conducted with serial production line configurations of different total buffer capacities and machines sizes. The experiments results of the proposed IACO and ACO algorithm are compared with those of DC algorithm presented in Ref. [11] and GA presented in Ref. [9]. In all the tests, it is assumed that the processing rate for each machine is one time unit; the values of the algorithm parameters are as follows: $\alpha = 1$, $\rho = 0.95$, $\gamma = 100$, $\beta = 15$. Table 1 shows the machine parameters of the production lines. All the experimental studies are programmed in C ++ language and run on a PC with Inter (R) Core (TM) 2-Duo (2.00GHz) CPU using the Windows7 operating system.

Table 1 Machines parameters

Machine	1	2	3	4	5	6	7	8	9	10
MTBF	20	20	30	22	30	30	20	25	26	30
MTTR	7	10	7	5	5	8	8	9	10	6
Machine	11	12	13	14	15	16	17	18	19	20
MTBF	10	30	30	20	25	45	10	20	12	25
MTTR	3	15	14	9	5	10	4	4	3	7
Machine	21	22	23	24	25	26	27	28	29	30
MTBF	20	25	25	40	30	20	30	22	22	25
MTTR	6	6	8	9	10	7	7	5	10	9

Table 2 presents the throughput rate, CPU time and number of evaluated solutions by using different optimization methods with k machines ($5 \leqslant k \leqslant 30$). It is shown that the IACO algorithm results in slightly larger throughput rate and requires less time and fewer numbers of evaluated solutions to find the near optimal solutions compared with other optimization methods in all of cases.

Table 2 Computational results by different optimization methods

k	N	Optimization method	E	CPU time(s)	Evaluated solutions
5	120	DC	0.648247	148.855	7000
		ACO	0.648406	3.252	315
		IACO	0.648617	0.889	150
		GA	0.648429	8.932	896
10	270	DC	0.640968	553.755	17000
		ACO	0.641198	68.181	2960
		IACO	0.64131	18.548	1200
		GA	0.641123	103.414	3168
15	420	DC	0.626742	748.211	22000
		ACO	0.626836	413.683	12430
		IACO	0.626887	153.973	4680
		GA	0.626796	380.865	5929
20	400	DC	0.603021	1732.4	32000
		ACO	0.603172	911.905	19040
		IACO	0.603229	689.433	10280
		GA	0.603178	993.873	14880
25	430	DC	0.595873	2750.6	34000
		ACO	0.595997	1838.51	25600
		IACO	0.596177	1218.19	17300
		GA	0.596095	1595.83	20800
30	590	DC	0.602541	3929.98	40000
		ACO	0.604243	2979.46	30600
		IACO	0.606567	1596.45	24200
		GA	0.604392	3028.17	29100

Fig. 4 shows the convergence curves of the four algorithms for the production lines with five, fifteen and thirty machines.

It is shown that the throughput rate of near optimal solution increases as the number of evaluated solutions increases for all of the DC, ACO, IACO and GA optimization methods. In addition, the proposed IACO algorithm results in solution quality higher and numbers of evaluated solutions fewer than the other three methods.

For the same number of machines, the different total buffer capacity will lead to a different number of evaluated solutions. Fig. 5 shows the effect of the N total buffer capacity to allocate among the production line on the number of the evaluated configurations needed

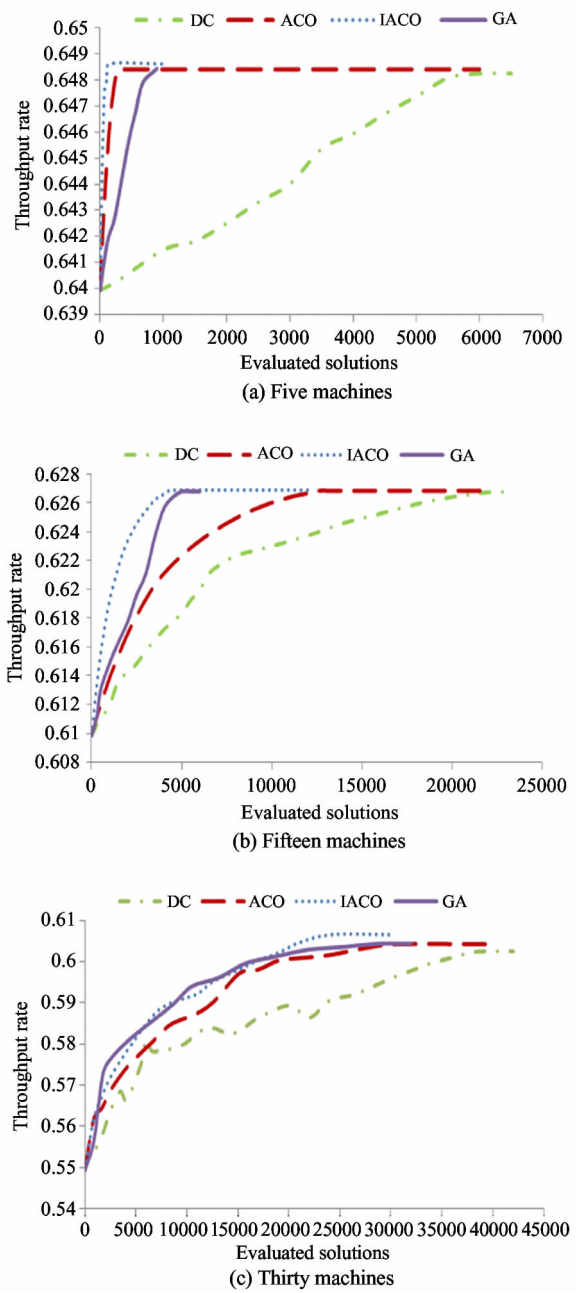

Fig. 4 Evolution of the solutions with four methods

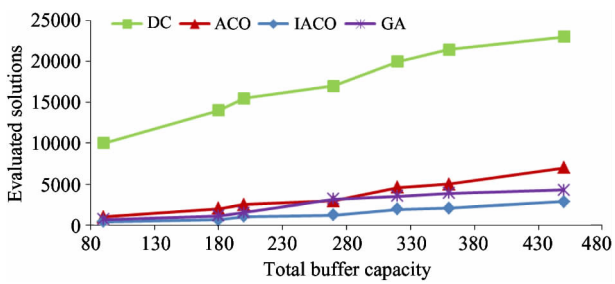

Fig. 5 Total buffer capacity versus number of evaluated solutions

to converge. This test focuses on a production line with ten machines, and the total buffer capacity varies from 90 to 450.

As shown in Fig. 5, the number of evaluated solutions needed by the IACO algorithm is lower than those of the DC, ACO and GA algorithm. In addition, the increase of the total buffer capacity leads to an increase of the number of evaluated solutions for near optimal buffer solutions.

As shown in Fig. 6, for six production lines with different sizes (from 5 to 30 machines) and different total buffer capacities, the number of evaluated solutions is increasing as the number of machines increases. In general, the IACO algorithm finds the best configuration much faster than the DC, ACO and GA algorithm.

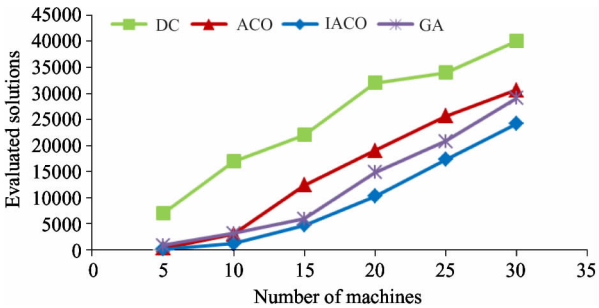

Fig. 6 Influence of the number of machines

Through the experiments analyzed above, it is easily known that the different number of machines and total buffer capacities will lead to a different throughput rate for the production line. As shown in Fig. 7, the throughput rate increases with the increase of the total buffer capacity when the number of machines is invariable. However, when the total buffer capacity is invariable, the throughput rate decreases with the increase of the number of machines.

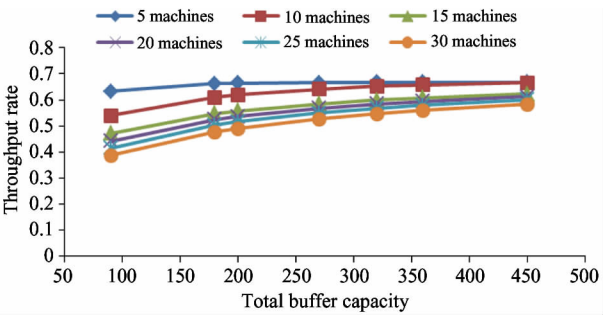

Fig. 7 Total buffer capacity and number of machines versus throughput rate

4 Conclusion

In this paper, an IACO algorithm is proposed to solve the BAP with the objective of maximizing the throughput rate in serial production lines with unreliable machines. For the ACO algorithm, some improvements are done to prevent it from local optimization, such as the two-opt strategy and the acceptance probability rule. Comparisons with other widely recognized methods are made to demonstrate the efficiency of our method. The results indicate that the IACO algorithm finds the optimal buffer configuration much faster than the other three approaches for different sizes production lines. Our future work will also test this approach on similar problems especially involving parallel machines.

References

[1] Spinellis D D, Papadopoulos C T. A simulated annealing approach for buffer allocation in reliable production lines. *Annals of Operations Research*, 2000, 93(1-4): 373-384

[2] Papadopoulos C T, O'Kelly M E J, Tsadiras A K. A DSS for the buffer allocation of production lines based on a comparative evaluation of a set of search algorithms. *International Journal of Production Research*, 2013, 51(14): 4175-4199

[3] Tsadiras A K, Papadopoulos C T, O'Kelly M E J. An artificial neural network based decision support system for solving the buffer allocation problem in reliable production lines. *Computers & Industrial Engineering*, 2013, 66(4): 1150-1162

[4] Ye T, Han W. A quantitative method to determine the size of the stock buffer in front of the bottleneck under multi-product. In: Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA), Dalian, China. 2006. 7239-7243

[5] Battini D, Persona A, Regattieri A. Buffer size design linked to reliability performance: A simulative study. *Computers & Industrial Engineering*, 2009, 56(4): 1633-1641

[6] Alexandros D C, Chrissoleon P T. Exact analysis of a two-workstation one-buffer flow line with parallel unreliable machines. *European Journal of Operational Research*, 2009, 197(2): 572-580

[7] Liu J, Yan D, Feng R, et al. Performance evaluation of production lines with unreliable buffer. In: Proceedings of the 8th IEEE International Conference on Control and Automation (ICCA), Xiamen, China, 2010. 350-355

[8] Massim Y, Yalaoui F, Amodeo L, et al. Efficient combined immune-decomposition algorithm for optimal buffer allocation in production lines for throughput and profit maximization. *Computers & Operations Research*, 2010, 37(4): 611-620

[9] Jeong S J, Jung H. Optimal buffer allocation in flexible manufacturing systems using genetic algorithm and simulation. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, 2012, 6(7): 1071-1080

[10] Sörensen K, Janssens G K. Simulation results on buffer allocation in a continuous flow transfer line with three unreliable machines. *Advances in Production Engineering & Management*, 2011, 6(1):15-26

[11] Nahas N, Ait-Kadi D, Nourelfath M. A new approach for buffer allocation in unreliable production lines. *International Journal of Production Economics*, 2006, 103(2): 873-881

[12] Liao C J, Tsai Y L, Chao C W. An ant colony optimization algorithm for setup coordination in a two-stage production system. *Applied Soft Computing*, 2011, 11(8): 4521-4529

[13] Akpinar S, Bayhan G M. Performance evaluation of ant colony optimization-based solution strategies on the mixed-model assembly line balancing problem. *Engineering Optimization*, 2014, 46(6): 842-862

[14] Zheng Q X, Li M, Li Y X, et al. Station ant colony optimization for the type 2 assembly line balancing problem. *International Journal of Advanced Manufacturing Technology*, 2013, 66(9-12): 1859-1870

[15] Dallery Y, David R, Xie X L. Approximate analysis of transfer lines with unreliable machines and finite buffers. *IEEE Transactions on Automatic Control*, 1989, 34(9): 943-953

[16] Kalayci C B, Gupta S M. Ant colony optimization for sequence-dependent disassembly line balancing problem. *Journal of Manufacturing Technology Management*, 2013, 24(3): 413-427

Zhou Binghai, born in 1965. He received his M. S. and Ph. D degrees respectively from School of Mechanical Engineering, Shanghai Jiaotong University in 1992 and 2001. He is a professor, a Ph. D supervisor in School of Mechanical Engineering, Tongji University. He is the author of more than 140 scientific papers. His current research interests are scheduling, modeling, simulation and control for manufacturing systems, integrated manufacturing technology.