Vol.2011-MPS-84 No.13
2011/7/18

部品装着におけるラインバランシング問題のための発見的解法

--- 装着順序が所与の場合 ---

戸崎博重[†]　太田秀典[†]　中森眞理雄[†]

基板に部品を装着する工程は，複数の装着機を連結したラインで行われることが多い．このラインの性能は，最も時間を要する機械で決るので，処理時間が均等になるように部品を装着機に割り当てることが必須である．しかし，各装着機における装着時間を見積ることは極めて難しい．筆者らは，各装着機に部品が割り当てられ，それらの装着順序も決定済みである場合に，装着時間を見積る新しい方法を考案した．その方法と旧来の方法に基づく装着時間を計算機実験によって求め，比較した結果を報告する．

A Heuristic Line Balancing Algorithm Accounting for Component Mounting Order

Hiroshige Tozaki[†], Hidenori Ohta[†] and Mario Nakamori[†]

A printed circuit board production line is composed of several component-mounting machines arranged in series that mount components onto each board. Since the performance of the line is determined by the most time consuming machine, the line balancing problem occurs that makes the mounting time by machines as equal as possible. Conventional line balancing procedures use time estimates based on the total number of components to be mounted under each machine. The actual mounting time, however, is often quite different from such estimates, and a substantial discrepancy arises between the actual production time and the estimated one of the line. In the present paper, a new estimation method of mounting time is proposed based on the calculation of the length of mounting paths, and also a heuristic algorithm of the line balancing problem is proposed. The proposed algorithm is shown by computer experiments to provide better results than conventional procedures.

1. Introduction

The present paper addresses the problem of allocating components in a component mounting line composed of machines. The target problem of allocating components in a component mounting line is to assign multiple components to each machine such that the production efficiency is the maximum. Both this problem and that of determining the component mounting order in each machine have influence to each other. As a result, even if each is optimized independently, the results will not necessarily result in overall optimization when combined. The allocation of components to each machine, however, has been optimized without taking the issue of component mounting order into consideration. Furthermore, estimation of the mounting time has been based on the component numbers of the components to be allocated by each machine [2], and no procedure accounting for the mounting path in each machine has been considered. Estimation methods based on the number of components show large discrepancies with actual production time. In the present paper we propose a heuristic algorithm that allocates components while it calculates the mounting paths in each machine.

2. Formulation of the Problem

Each component mounting machine contains a head equipped with multiple vacuum nozzles that pick components and mount them on the board. The action of a machine is as follows: Using its vacuum nozzles, the head first picks up multiple components that have been positioned on the supply feeder, transports them to the location on the board where one of the components is to be mounted, mounts one component, then moves to the location where a second component is needed and mounts that component and so on, repeating until all of the components it picked up have been mounted, after which the head returns to the supply feeder to pick up another batch of components. The sequence of actions from the pick-up to the return of the head to the supply tray is called a turn.

The present paper addresses the component allocation problem while at the same time taking into consideration the mounting paths of each machine. A line is assessed based on the time spent per single board by the bottleneck machine. This time is strongly affected by the travel distance of the head, which is defined as the Chebyshev distance because head is driven by motors that operate independently in the x and y directions. Therefore, our algorithm calculates the head travel distance for each machine considering both component

[†] 東京農工大学
Tokyo University of Agriculture and Technology

allocation and component mounting order, evaluating the component allocation by the maximum head travel distance.

Since this paper focuses on component allocation and component mounting order, the problem of component pick-up is only briefly addressed; all components are assumed to be picked up at an identical point on the supply feeder; the distances between nozzles are also neglected by assuming that the head makes no extra motions to pick up components. For the sake of simplicity, it is also assumed that all the components could be picked up by any nozzle, and that there is no upper limit on the number of component types that could be placed on the supply feeder.

The above problem is formulated as an integer programming problem as follows:

$$\text{minimize} \quad z = \max(T_j) \quad j = 1, \ldots, jsize \qquad (1)$$

subject to

$$T_j = \sum_{r=1}^{m_j} \left\{ t(c(0), c(m_{jr})) + \sum_{i=1}^{m_{jr}} t(c(i-1), c(i)) \right\} \qquad (2)$$

$$m_{jr} \le N \qquad (3)$$

$$t(a,b) = \max(|x_b - x_a|, |y_b - y_a|) \qquad (4)$$

$$M = \sum_{j=1}^{jsize} \sum_{r=1}^{m_j} m_{jr} \qquad (5)$$

z : objective function

T_j : total travel distance of machine j

$jsize$: total number of machines comosing the production line

m_j : number of component mounting paths in machine j

$c(i)$: coordinates of i th compoonent on the mounting path

$c(0)$: coordinates of the supply feeder

m_{jr} : total number of components included in path r of machine j

N : number of nozzles

x_a : x coordinate of component a

y_a : y coordinate of component a

M : total number of components to be mounted

Formula (1) is the objective function and states that the maximum value of the total head travel distance of each machine calculated in (2) should be minimized. Constraint (3) limits the maximum number of components that can be mounted in one turn. Constraint (4) defines the Chebyshev distance between two component insertion points. Constraint (5) expresses the number of components mounted on a single board.

3. Conventional Procedures

In conventional procedures, once the components have been allocated to machines in a balanced manner according to the number of components, the mounting order of the components in each machine is determined and the head path is created. This procedure is easily adaptable to simulated annealing as follows: we use the allocation obtained from the above conventional method as the initial solution; in order to improve temporary solution, we exchange the components allocated to the bottleneck machine with those allocated to other machines or pass the components allocated to the bottleneck machine to another machine. We call this method “extended conventional procedure.” Figure 1 shows a flow diagram for the extended conventional procedure.

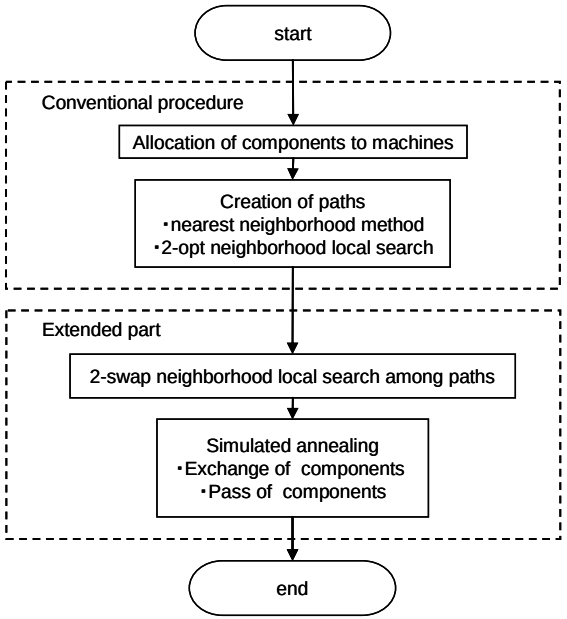

Figure 1 Extended conventional procedure

4. Proposed Algorithm

In the present paper we propose a hybrid genetic algorithm, which is a combined version of a genetic algorithm with local search technique. Figure 2 shows a flow diagram of the proposed algorithm.

Incorporating local search technique into the genetic algorithm strengthens weak points of genetic algorithm and provides a more precise solution. In the algorithm proposed here, local search is used for initial solution generation, mutation and offspring solution improvement.

4.1 Expression of solutions

In the proposed algorithm, all machines are assumed to mount their components in the same number of turns. When the number of turns increases, the head must move between the supply tray and the board more frequently, so it is desirable that the number of turns for each machine be reduced as much as possible. The minimum number of turns necessary for a single machine is obtained by (6). Note that, depending on the number of machines and components available for mounting, there will be turns during which no components are picked up by some of the nozzles.

$$m_j = \left\lceil \frac{M}{N \times jsize} \right\rceil \qquad (6)$$

The components are assigned individual numbers, and the solution is expressed by a row of these component numbers aligned as shown in Fig. 3. If the arranged components are separated into blocks based on a fixed number of components starting from the first, this will correspond with their allocation to the machines and paths. Furthermore, the sequence of component numbers within a turn will reflect the mounting order. Figure 3 shows an example involving two machines and six nozzles, where each machine carries out two turns of component mounting. Unused nozzles are flagged with the letter “e” in place of the component number. Figure 4 shows the component allocation corresponding to the solution shown in Figure 3 and the paths of the head.

4.2 Fitness of solution

The evaluation value is the head travel distance of the bottleneck machine. Accordingly, when we select parents and individuals during crossover based on the score, we are not considering any machines other than the one that has become the bottleneck. However, the path lengths in the other machines might have a significant influence on the solution generated by crossover and mutation. Therefore, in this paper, in addition to the interconnect length in the machine that is identified as the bottleneck, we calculate the fitness value of the solution while incorporating the total interconnect length for all machine heads. The fitness value of each individual is the inverse of the square of the sum of the head travel distance of the bottleneck machine and the head travel distances of all the other machines. Equation (7) shows how to calculate the total path length L and Equation (8) shows how to calculate the fitness value f, where S is the size of the area.

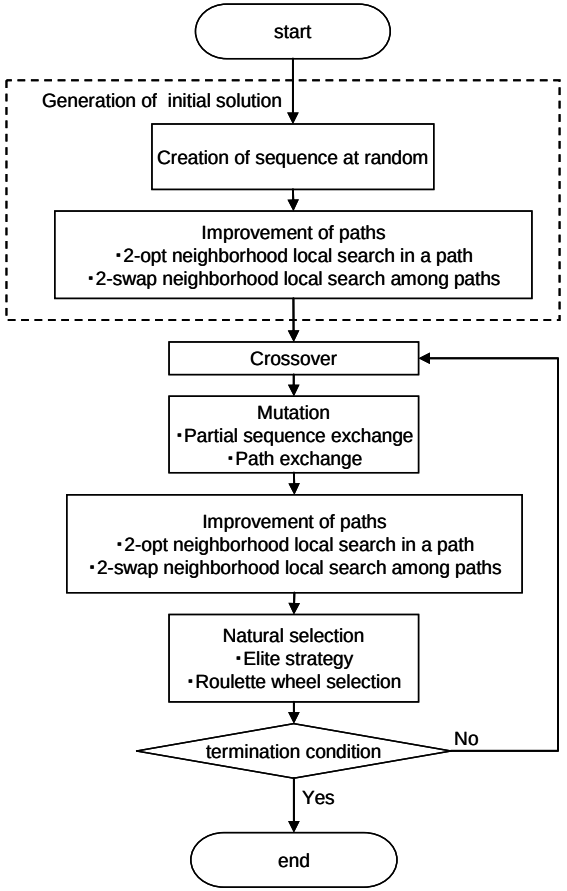

Figure 2 Proposed algorithm

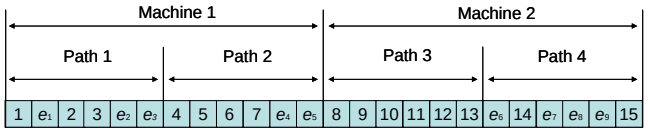

Figure 3 Expression of the solution

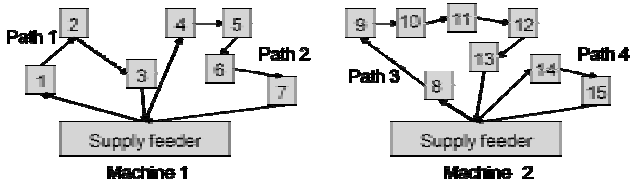

Figure 4 Component allocation corresponding to Fig 3

$$L = \sum_{j=1}^{jsize} T_j \tag{7}$$

$$f = \left(\frac{1}{z + L} \right)^2 \times S \tag{8}$$

4.3 Generation of the initial solution

Our genetic algorithm requires initial solutions as many as the pre-determined number of individuals. Each of the initial solutions is generated using the following two steps:

(1) A sequence is created at random.

(2) Sequences are improved by 2-opt and 2-swap neighborhood local search.

4.4 Crossover

Crossover is performed in the proposed algorithm in the order from (1) to (3) below, to obtain a child:

(1) A partial sequence of random length is selected from parent 1.

(2) A sequence is formed by removing all of the elements included in the partial sequence selected in step (1) from parent 2.

(3) The partial sequence selected in step (1) from parent 1 is inserted into the sequence created in step (2) from parent 2. The insertion location is the same location at which the selected sequence had previously existed in parent 1.

Figure 5 shows an example of crossover. For two parent individuals, crossovers are performed that are randomly selected at a probability proportional to their fitness values. The above steps (1)-(3) are executed for parents 1 and parent 2 as well as for parent 2 and parent 1, so we have two children from one couple.

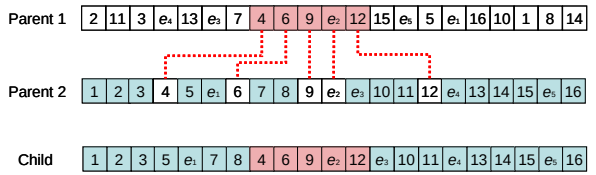

Figure 5 Crossover

4.5 Mutation

Two types of mutations are proposed here, partial sequence exchange and path exchange. The parent individuals used to generate the mutation are selected at random from all the individuals.

(1) Mutation by partial sequence exchange

In mutation by partial sequence exchange, two partial sequences are selected at random from each selected individual and exchanged. The length of the partial sequence is changed at random within a specified range.

(2) Mutation by partial sequence exchange

In mutation by path exchange, two partial sequences corresponding to paths are selected at random from each selected individual and exchanged.

4.6 Natural selection

30% of the present generation are selected by the elite strategy as individuals left to the next generation. Some of the remaining 70% unselected by the elite strategy are decided by roulette wheel selection to be individuals left to the next generation. In roulette wheel selection, individuals are chosen at a probability proportional to fitness.

5. Computer Experiment

We made a comparative experiment on a computer to evaluate the performance of the proposed procedure. The computer environment is an Intel Core2 Duo 3.00 GHz CPU with 1.96 GB of memory. The programming language used was C++.

The supposed line contained six machines, each with a head equipped with 12 nozzles.

The starting point for insertion was at a location 350 mm in the y direction from the center of gravity of the board. Nine sets of data were prepared for entry, which randomly specified the coordinates of the mounting locations on a board 100 mm wide by 100 mm deep in size.

The proposed algorithm used 25 individuals and the ending condition was the 3000th generation. The values for other parameters were specified appropriately based on a preliminary experiment. Figure 8 shows the relation between the scores for the proposed procedure and the extended conventional procedure as well as the calculation time and shows that the proposed procedure quickly obtains better solutions than the extended conventional one.

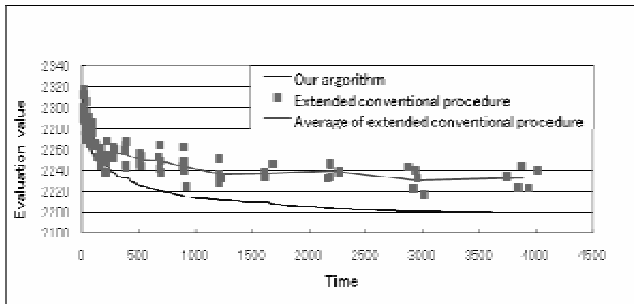

Figure 8 Evaluation value and time

Table 1 presents the evaluation values for the proposed procedure, the conventional procedure, and the extended conventional procedure obtained after a sufficient length of time, for each data set entered. The proposed procedure obtained the best evaluation values in all of the nine data sets of various types. The extended conventional procedure showed a mean improvement over the conventional procedure of 13.9%, while the proposed procedure showed a mean improvement over the conventional procedure of 14.6%.

6. Conclusions

In this paper we proposed a procedure for allocating components to machines on a printed circuit board production line by actually creating component mounting paths, rather than by creating rough estimates using only the number of components to be mounted. Computer experiment shows that the proposed procedure provides good solutions than the conventional procedure or an extended version of the conventional procedure. Issues to be addressed in the future include incorporating limitations specific to printed circuit board production lines, such as number of component types and component heights.

Table 1 Comparison of conventional procedure, extended conventional procedure and our algorithm

		Conventional procedure		Extended conventional procedure		Our algorithm	
	Number of mounting points	Search time (sec)	Evaluation value	Search time (sec)	Evaluation value	Search time (sec)	Evaluation value
1	100	0.005	1694.93	837.08	1420.66	495.92	1414.69
2	100	–	1674.24	840.00	1405.06	525.12	1404.95
3	100	–	1675.75	829.32	1441.25	520.31	1423.67
4	200	0.005	2594.38	3896.77	2232.47	3632.48	2197.78
5	200	0.016	2594.04	3922.46	2225.78	3648.23	2194.31
6	200	–	2585.95	3880.22	2210.18	3869.80	2181.23
7	400	0.130	5048.42	42167.83	4381.65	34179.47	4357.23
8	400	0.125	5041.57	39346.17	4381.29	31826.50	4360.36
9	400	0.130	5052.73	38433.20	4373.88	31012.80	4342.90

7. References

[1] Masri Ayob, Graham Kendall. A survey of surface mount device placement machine optimisation: Machine classification. European Journal of Operational Research 186 (2008) , 893–914

[2] Osman Kulak, Ihsan Onur Yilmaz. Hans-Otto Günther, A GA-based solution approach for balancing printed circuit board assembly lines. OR Spectrum 30 (2008), 469–491

[3] Sun, D.S., Lee, T.E., Kim, K.H.: Component Allocation and Feeder Arrangement for a Dual-Gantry Multi-Head Surface Mounting Placement Tool. International Journal of Production Economics, Vol.95, pp.245?264 (2005).

[4] Yamada, T., Miyashiro, R., and Nakamori, M.: An Algorithm of Feeder Arrangement and Pick up Sequencing of Component Placement Machine on Printed Circuit Board, Proc. International Conference on Parallel and Distributed Processing Techniques and Applications, pp.403-409 (2005).

Vol.2011-MPS-84 No.13
2011/7/18

[5] Ahmadi, R. H. and Mamer, J. W.: Routing Heuristics for Automated Pick and Place Machines, European Journal of Operational Research, Vol. 117, pp. 533?552 (1999).
[6] Burke, E. K., Cowling, P. I. and Keuthen, R.: New Models and Heuristics for Component Placement in Printed Circuit Board Assembly, International Conference on Information Intelligence and Systems, pp. 133?140 (1999).
[7] Burke, E. K., Cowling, P. I. and Keuthen, R.: Effective Heuristic and Metaheuristic Approaches to Optimize Component Placement in Printed Circuit Board Assembly, Evolutionary Computation 2000 Proceedings of the 2000 Congress, pp. 301?308 (2000).
[8] Burke, E. K., Cowling, P. I. and Keuthen, R.: The Printed Circuit Board Assembly Problem : Heuristic Approaches for Multi-Headed Placement Machinery, Proceedings of the International Conference on Arti?cial Intelligence ICAI’2001 , pp. 1456?1462 (2001).
[9] Hackman, S. T., Magazine, M. J. and Wee, T. S.: Fast, Effective Algorithms for Simple Assembly Line Balancing Problems, Operations Research, Vol. 37, pp. 916?924 (1989).