ELSEVIER

Int. J. Production Economics 64 (2000) 399–407

international journal of production economics

www.elsevier.com/locate/dsw

The feeder rack assignment problem in PCB assembly: A case study

Cornelis Klomp[a], Joris van de Klundert[b], Frits C.R. Spieksma[b,*], Siem Voogt[a]

[a]*Philips Centre for Manufacturing Technology, Building SAQ-2, P.O. Box 218, NL-5600 MD Eindhoven, The Netherlands*
[b]*Department of Mathematics, Maastricht University, P.O. Box 616, NL-6200 MD Maastricht, The Netherlands*

Abstract

In this paper we describe a case study in the operational planning of the assembly of printed circuit boards. Given a line of placement machines and given a family of boards we develop a heuristic algorithm which focusses on the so-called *feeder rack assignment problem*. This problem addresses the question: where to attach the feeders alongside the feeder racks of the placement machines? Computational results on real-life instances indicate that the heuristic is superior to approaches currently used in practice. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Heuristics; PCB-assembly; Feeder rack assignment problem

1. Introduction

Assembling a printed circuit board (PCB) means that expensive, complex machines place hundreds of electronical components of different types on a board. To illustrate this, it is estimated in [1] that a typical plant in the PCB-industry has 29 placement machines with a total value of at least 1.5 million dollars. In order to use these machines efficiently (so as to achieve high throughput rates), automated planning procedures are generally regarded as an asset, or even a necessity.

Here, we describe a case study in the planning of the assembly of printed circuit boards. This case study has been conducted in a plant of Philips NV. We focus on the operational planning problems which occur when a *given* line of placement machines assembles a *given* family of board types.

Before describing these problems in more detail, let us first sketch the modus operandi of the placement machine under consideration, which is a Fuji CP IV/3 (see Fig. 1). Basically, this placement machine consists of three elements:

- A PCB table. The PCB lies on this table which is able to move horizontally as well as vertically.
- A feeder rack. The feeder rack consists of the so-called *slots*. A rack contains feeders each of which stores components of a single type and which are attached to the slots. A feeder occupies a single slot.
- A carousel. This is a device which is able to transport components from the rack to the board.

* Corresponding author. Tel.: + 31-43-3883359; fax: + 31-43-3211889.
E-mail address: spieksma@math.unimaas.nl (F.C.R. Spieksma)

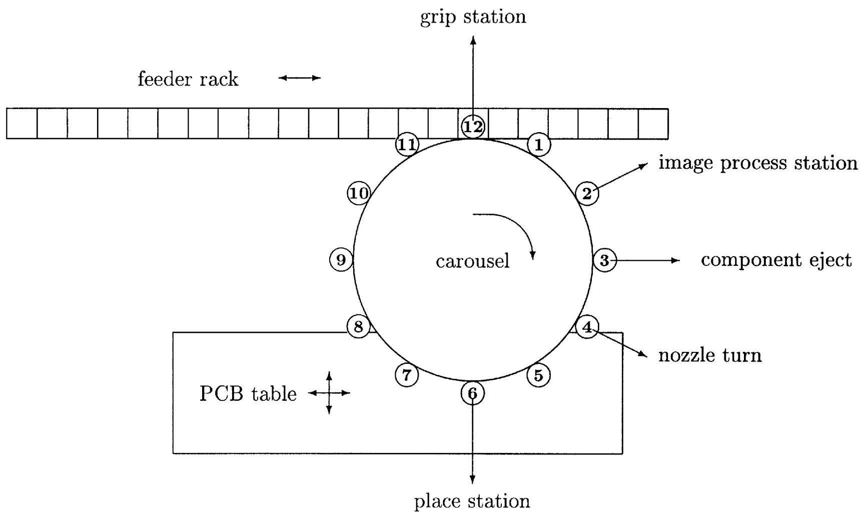

Fig. 1. The Fuji CP IV/3.

Now, the machine operates as follows (see Fig. 1). The head in the grip station is able to grip a component from the feeder currently positioned under this head. Then, the feeder rack moves to get the next appropriate feeder in position. In a similar fashion, the head in the place position is able to place a component into the location of the board which is currently under this head. Then the board starts to move to get the next location under the place station. Moreover, when the gripping activity and the placing activity are both ended the carousel rotates 30 degrees, so that head i comes into position of head $i+1$ (modulo 12). Then a new iteration starts. It is important to realize that the description above not only suits placement machines from the Fuji CP family; in fact different types of placement machines (see [2]) share the characteristics described above.

It follows from this description that between two consecutive gripping (or placing) activities a carousel movement must take place. The time needed for this movement, say Δt, can be used to move the feeder rack and the PCB table without adding to the processing time of the current board. More explicitly, any feeder rack movement and any table movement which takes less time than Δt is for free. With respect to the feeder rack this amounts to 1 slot in the rack, and with respect to the PCB table this amounts to approximately 10 cm (see [3] for a more precise description of time-related issues). It is clear that in general, due to the absence of some component types on some board types, it will not be possible to construct a feeder rack assignment such that all feeder rack movements for all board types are for free. Thus, the positions of the feeders in the racks determine to a large extent the throughput rates of the board types.

Before any placement machine of a type as described above can actually start placing components, or even more generally, before a line of placement machines is able to assemble a family of board types, a number of operational questions have to be answered. For instance:

- For each component type, how many feeders of this type should be present in the line? (The selection problem.)
- For each machine, which feeders should be attached to the feeder rack? (The allocation problem.)
- For each machine, where (in what slots) should the selected feeders be attached? (The feeder rack assignment problem.)
- For each machine and for each board type, in what sequence should the locations be visited? (The component placement sequence problem.)
- For each machine and for each board type: if more than one feeder of some type is present in

the rack, which feeder should be selected to retrieve a particular component of that type from? (The component retrieval problem.)

The questions listed here already suggest a hierarchical decomposition of this operational planning problem. But of course different possibilities exist to decompose the planning problem (see [2,4–7] for a discussion of such decompositions). However, no matter what hierarchical decomposition is chosen, and regardless of the particular type of placement machine, the feeder rack assignment problem should always be solved. Moreover, it seems to be a basic problem arising in any hierarchical decomposition.

It should be pointed out though, that the relevance of the feeder rack assignment problem stems from the assumption that the location of the feeders has an impact on the objective function, say the throughput. Although this seems quite reasonable, one can imagine situations in which the particular location of a feeder in the rack is assumed to have no impact on the time it takes to assemble a board (see for instance [8,9]).

However, in the particular setting we address here, the production environment is a high-volume, medium-mix environment. A typical situation is that a single line of placement machines produces a family of up to 20 board types during a month. In this period no feeder changes occur (other than for refilling purposes). Thus, when dealing with throughput rate objectives the solution to the feeder rack assignment problem is crucial for the performance of the line.

1.1. Related literature

Literature on the feeder rack assignment problem has mainly focused on the single-machine, single-board case. Drezner and Nof [10] were the first to identify the feeder rack assignment problem. Walas and Askin [11] considered a similar problem in a metal-cutting environment. In the single-machine, single-board-type case it is possible to combine the feeder rack assignment problem with the component placement sequence problem. The resulting problem can be formulated as a (nonlinear) integer programming problem (see [11]). Next, a popular approach is to develop a solution method which iterates between the feeder rack assignment problem and the component placement sequence problem. This is done by Leipälä and Nevalainen [12] and by Broad et al. [13] for the PANASERT machine. Another approach of this type is presented in [14,15].

Ahmadi et al. [16] consider the feeder rack assignment problem (which they call the *reel positioning problem*) for the DYNAPERT placement machine. They do this *given* a component placement sequence and show that for their setting the feeder rack assignment problem is NP-hard and provide an approximation algorithm with worst-case ratio $\frac{3}{2}$. Other approaches which use a component placement sequence as input are described in [17] and in [18] for the PANASONIC Mk 1 (which is similar to the machine described in Fig. 1). In [19] the Fuji CP II is addressed. They compute a feeder rack assignment (which they call a component assignment) by solving an integer program using Lagrangian relaxation. The feeder rack assignment problem for the so-called CSM-60 placement machine is adressed in [20] and in [21]. Heuristics are used in the first one (an iterative approach), simulated annealing in the latter one. Finally, approaches based on location theory are described by Francis et al. [22] (for an IBM placement machine), Foulds and Hamacher [23] and Younis and Cavalier [24]. In [25] it is shown that the algorithm proposed in Francis et al. [22] has a worst-case ratio of $\frac{3}{2}$.

Literature describing solution methods for the feeder rack assignment problem in a multiple-board type setting seems less abundant. The problem is mentioned in [26]. Dikos et al. [27] compute a feeder rack assignment for the PANASONIC machine given component placement sequences for each board type using a genetic algorithm. In [28] a heuristic is proposed. Further, this problem is specifically addressed in Crama et al. [20] who consider the Fuji placement machine.

An important issue is how often the feeder rack assignment problem should be solved. Should one solve it at the beginning of the planning period and leave the resulting solution intact for the whole period for all machines (as is the case in our setting)? Or should one first assign the feeders corresponding to common component types, place these components, and next perform a setup for the feeders corresponding to the remaining component types, as proposed in Carmon et al. [30]? Or should a part of the rack be permanently reserved for the common feeders while remaining capacity is changed more often (see [31])? There are no clear-cut answers to these questions. These decisions depend to a large extent on the machine configuration and the production environment in the particular plant (see [32] for an interesting discussion of this subject).

1.2. Contents

Crama et al. [29] proposed an algorithm for solving the problems mentioned above, starting with the selection problem down to the component retrieval problem. However, it turned out that when implementing this algorithm in a Philips plant, some additional requirements came up. These requirements affected the selection problem and the feeder rack assignment problem, so our focus in this paper is on both of these problems. The purpose of this paper is twofold:

- to show how the approach presented in [29] can be modified so as to incorporate additional constraints, and
- to present new computational results that demonstrate the potential of our approach.

In Section 2 we describe our algorithm and the modifications proposed, and in Section 3 we give the computational results. Section 4 presents the conclusions.

2. The algorithm

As described in the previous section we consider a high-volume, medium-mix production environment. More explicitly, during the planning period (which is about a month) the board types of a family are produced in a predetermined sequence using a constant feeder rack assignment. Thus, achieving high throughput rates for the boards of a type is of crucial importance. Obviously, the throughput rate of a board type is determined by the machine in the line on which a board of that type has the largest processing time over all machines. We refer to this largest processing time as the *makespan* of that board type. When assembling a set of board types using a line of placement machines, a first question to deal with concerns the objective function. What do we want to achieve? In light of the previous discussion we choose to minimize the sum over all board types of the makespans of these board types. Thus, we want to construct a single feeder rack assignment for all machines in the line that allows good component placement sequences for *all* board types in the family.

The following algorithm is described in [29] (see [29] for a more elaborate description). Consider the selection problem. First, it is stipulated that no more than two feeders of a type are present in the line. Then the remaining decision is: for each type, does it have one or two feeders in the line? We solve this in a heuristic fashion by considering individual board type characteristics. (Notice here that having more than one feeder of a type present in the line, i.e. *feeder duplication*, is allowed).

Let us now decide which feeder serves which locations on the board. Obviously, if for a component type only one feeder is present in the line, this becomes a trivial decision. Otherwise, we use a heuristic to split into two sets the locations of each component type on each board type. We refer to these sets of locations served by a single feeder as a *cluster*. (Recall that the feeders have not been placed yet.)

To assign the feeders to the racks of the machines we do the following (see [29]). First, we view the racks of the individual machines in the line as a single, large rack. Also, let us view a feeder (and its corresponding cluster) as a node in a complete graph. For each pair of nodes (feeders), say i and j, we compute the length of edge $\{i, j\}$ in the following way. Find, for each board type in the family, a Hamiltonian path (as short as possible) through the locations of the clusters corresponding to feeders i and j. We do this by applying an insertion algorithm. (Let us remark here that when computing the length of such a Hamiltonian path, a lower bound for the time between any two locations is formed by the time needed for a carousel movement). The length of edge $\{i, j\}$ is then the sum, over all board types, of the lengths of these Hamiltonian paths. In fact, this length is intended as an approximation of the contribution to the processing times of the board types when feeder i and are j placed consecutively on the rack of a machine. The problem of finding a feeder rack assignment has now been reduced to finding a Hamiltonian path in a graph. Again, we solve this problem by applying an insertion algorithm. Next, we simply partition the path into disjoint subpaths which then correspond to the racks of the individual machines.

An important feature of the algorithm ([29]) is that *given* a feeder rack assignment it is able to compute an estimate of the makespan of each board type. This estimate makes use of the Hamiltonian paths described above. Consider machine 1 and board type A and sum over all consecutive feeders in the rack of machine 1 the lengths of the corresponding Hamiltonian paths for board type A, and divide this sum by 2. The resulting number is used as an estimate for the processing time of board type A on machine 1. From these estimated processing times, an estimate for the makespan is readily deduced.

When computing this estimate we simply assume that between two consecutive carousel movements, the feeder rack moves by a single slot from left to right only. In this way, each feeder rack movement takes place *within* the time needed for a carousel movement and hence is for free.

Finally, we apply a local search algorithm that swaps feeders and their corresponding clusters. In this way we are able to obtain a more balanced workload over the machines.

When we ran this algorithm in the Philips plant two issues came up: the concept of preassigned feeders (Section 2.1) and the issue of feeder duplication (Section 2.2).

2.1. Preassigned feeders

In the Philips plant feeders corresponding to expensive component types can be *preassigned*. In other words, they are required to be in certain predetermined slots of the rack of the last machine in the line. The reason for this is that in case something goes wrong with the placement process and the board is lost, it will most probably not contain the expensive components. What are the consequences of this issue for the algorithm?

As described above, our heuristic approach views feeder rack assignments as Hamiltonian paths. Thus, the problem of determining an optimal feeder rack assignment for some machine can be viewed as finding a shortest Hamiltonian path, for which we employ an insertion heuristic. And indeed, from the literature on the Traveling Salesman Problem (see [34]) it is well known that insertion heuristics are among the best constructive heuristics in finding short paths or tours. Unfortunately, the concept of preassigned feeders does not combine well with insertion heuristics. To see this, notice that when a new feeder is inserted at slot i in a feeder rack assignment, the feeder currently assigned to slot i is reassigned to the slot with index $i + 1$. Also, the feeder currently assigned to slot $i + 1$ moves to slot $i + 2$ and so on. It follows that this insertion approach is not suited to deal with situations where some feeders have fixed positions. Thus, for the feeder rack assignment problem with preassigned feeders we had to resort to a more primitive constructive heuristic to find an initial assignment. We have modified our algorithm so as to start with a random assignment of feeders to slots, except of course for the feeders of types for which there are preassigned feeders. In the solution of the selection problem we have ascertained that there is at least one feeder for each component type with a preassigned feeder, and hence in the initial feeder rack assignment the algorithm can and will assign a cluster containing components of the appropriate type to each of the preassigned feeders. The local search improvement strategy we now employ is to simply exchange feeders and their corresponding clusters if the estimated objective function value decreases. The potential feeder exchanges we allow are between pairs of feeders that are not preassigned, or pairs of preassigned feeders of a same type. Also, we do allow exchanges of feeders that are not preassigned with empty slots, that is repositioning a single feeder.

2.2. Feeder duplication

Another issue that turned out to be important was *feeder duplication*: the presence of more than one feeder of some type in the line. The possibility of ruling out feeder duplication is considered to be a necessary one, or in other words, an engineer at the Philips plant should have the option of constructing a feeder rack assignment that contains no feeder duplication. Again, this is caused by cost considerations: duplicating a feeder simply increases costs of assembly, since it requires more inventory. For a more elaborate discussion of the trade-off between throughput rates and inventory costs we refer to Rosenblatt and Lee [33]. However, this requirement can be simply incorporated: due to the no duplication requirement, we now have an immediate solution for the selection problem. Of course, an interesting question arises: how does ruling out feeder duplication affect the throughput rates? This issue is addressed in Section 3.

3. Computational results

This section presents computational results. For all these results it holds that either the results themselves or the software through which they are obtained have been validated by Philips. The algorithm we described is programmed in Borland Pascal and runs on a personal computer.

The computational results are carried out for placement lines consisting of 2 or 3 Fuji CP-IV/3 placement machines. Gripping as well as placing a component takes at least 150 milliseconds. Assuming that the printed circuit boards are assembled at this highest achievable placement rate yields a lowerbound for the assembly makespan. These lowerbounds are given in the tables below.

We use two data sets, one corresponding to a family of two board types (data set 1) and one corresponding to a family of 20 board types (data set 2); data set 1 concerns 873 components of 115 types and data set 2 concerns more than 8200 components of 123 types. Further, we consider two assembly lines, one consisting of two and one consisting of three machines. While the experiments were in progress, Philips was using two different versions of (extensions of) the software that Fuji delivers to solve operational planning problems. We compare our results with the best of these versions.

In our first experiment we tested the algorithm described here using dataset 1 on both lines. The results are depicted in Tables 1 and 2 (bold figures indicate the makespans), where NoC means number of components to be placed on a board of that type by that machine and time indicates the number of seconds it took that machine to place these components. Notice that adding a machine to the line improves the objective function by about 30%. Also notice that, particularly in the three machine case, the gap between the solution found and the lowerbound is relatively small (about 20%), which implies that a lot of table movements and rack movements fall within the free movement.

When the feeders corresponding to eight expensive components are preassigned, the algorithm finds the makespans depicted in Tables 3 and 4. Notice that in both cases the effect from the preassigned feeders on the makespan is bounded by approximately 2%.

In a second experiment, we tested the effect of feeder duplication on the makespans of board types within a family (data set 2). These results where

Table 1
Results data set 1, two machines

Board type	Machine 1 NoC time	Machine 2 NoC time	Lowerbound
1	240 42.8	223 **43.2**	35.6
2	213 **38.1**	197 37.6	31.7

Table 2
Results data set 1, three machines

Board type	Machine 1 NoC time	Machine 2 NoC time	Machine 3 NoC time	Lowerbound
1	159 **30.2**	142 29.1	162 29.3	24.1
2	140 **26.3**	129 25.7	141 26.2	21.4

Table 3
Results data set 1, preassigned feeders, two machines

Board type	Machine 1 NoC time	Machine 2 NoC time	Lowerbound
1	235 43.9	228 **44.1**	35.6
2	207 **38.9**	203 38.8	31.7

Table 4
Results data set 1, preassigned feeders, three machines

Board type	Machine 1 NoC time	Machine 2 NoC time	Machine 3 NoC time	Lowerbound
1	158 **30.2**	147 29.2	158 29.0	24.1
2	141 **26.6**	130 25.7	139 26.2	21.4

verified on site, and compared to both versions of the Fuji software in use at the time. Table 5 displays the makespans when no feeder duplication is allowed. We have omitted the number of components per board. The average makespan per board equals 42.2. The software on site delivers an average makespan per board of 49.2. Thus in case feeder duplication is allowed, our method yields an improvement of about 14% for data set 2 over the current solution. If feeder duplication is not allowed, the difference is even larger. The algorithm proposed here yields an average makespan of 43.0 seconds (see Table 6 for the processing times/board type/machine). Notice that this yields an increase of only 0.8 seconds when compared to the case where feeder duplication is allowed. (In fact, for board type 20, the makespan actually decreases!). In contrast to this, the software on site delivers a solutions with an average makespan of 55.4 seconds when feeder duplication is not allowed. Apparently, the currently implemented solution method is quite sensitive to feeder duplication, since an increase of about 7% in makespan is incurred when compared to the solution in which feeder duplication is allowed (and the latter solution was already 14% higher than the one found by the algorithm discussed here).

Table 5
Results data set 2, feeder duplication allowed

Board type	Machine 1	Machine 2	Machine 3	Lowerbound
1	45.2	**46.5**	45.4	23.1
2	**43.9**	43.1	43.7	22.7
3	40.2	40.4	**42.0**	21.2
4	41.5	**44.9**	43.0	22.9
5	41.5	**44.6**	43.7	22.8
6	40.6	**43.2**	40.7	21.9
7	45.9	**46.1**	45.8	24.1
8	40.8	41.8	**42.2**	21.4
9	24.7	24.4	**24.9**	12.6
10	42.0	41.2	**42.1**	21.9
11	39.0	**40.0**	**40.0**	20.9
12	40.1	**41.6**	41.0	21.3
13	42.5	42.8	**43.3**	22.3
14	44.6	**45.0**	43.6	23.1
15	**43.4**	43.1	**43.4**	22.5
16	43.0	**44.2**	43.8	22.4
17	42.7	43.4	**43.9**	22.3
18	40.9	40.9	**41.4**	22.1
19	43.2	44.2	**45.1**	22.9
20	**35.7**	34.2	35.0	17.4

Table 6
Results data set 2, no feeder duplication allowed

Board type	Machine 1	Machine 2	Machine 3	Lowerbound
1	45.9	**46.6**	44.7	23.1
2	43.6	44.7	**45.7**	22.7
3	39.9	**42.9**	41.0	21.2
4	**44.9**	44.8	44.7	22.9
5	**44.9**	44.8	44.3	22.8
6	**43.2**	43.1	42.3	21.9
7	**47.5**	46.5	46.9	24.1
8	**43.2**	40.8	**43.2**	21.4
9	25.2	**26.4**	23.6	12.6
10	42.6	**42.9**	42.4	21.9
11	39.7	**41.1**	40.1	20.9
12	41.5	**42.3**	42.2	21.3
13	**44.0**	43.6	42.8	22.3
14	44.5	**45.5**	45.0	23.1
15	43.0	44.2	**46.1**	22.5
16	42.6	**44.9**	43.1	22.4
17	42.6	**45.9**	43.1	22.3
18	40.0	**42.3**	39.6	22.1
19	**45.1**	44.8	44.7	22.9
20	33.5	34.1	**34.7**	17.4

4. Conclusions

In industries with thin profit margins such as consumer electronics (where the data sets used in this paper stem from), cost-effectiveness is of crucial importance to stay in competition. The results in this paper indicate that for different variations of the printed circuit board assembly problem, the use of clever heuristics may increase competitiveness significantly.

Acknowledgements

We would like to thank Yves Crama for his comments on an earlier version of this paper and Olaf Flippo for his role in the research which underlies this paper.

References

[1] A. Mody, R. Suri, M. Tatikonda, Keeping pace with change: International competition in printed circuit board assembly, Industrial and Corporate Change 4 (1992) 583–614.
[2] S. Grotzinger, Feeder assignment models for concurrent placement machines, IIE Transactions 24 (1992) 31–46.
[3] C.J.G. Hehl, Time model FUJI CP-IV/3, including cycle time measurements, Report of the Philips Centre for Manufacturing Technology, Eindhoven, 1994.
[4] R.H. Ahmadi, A hierarchical approach to design, planning, and control problems in electronic circuit card manufacturing, in: R.K. Sarin (Ed.), Perspectives in Operations Management, Kluwer Academic Publishers, Dordrecht, 1993, pp. 409–429.
[5] Y. Crama, A.G. Oerlemans, F.C.R. Spieksma, Production Planning in Automated Manufacturing, 2nd Edition, Springer, Berlin, 1996.
[6] M.M. Dessouky, S. Adiga, K. Park, Design and scheduling of flexible assembly lines for printed circuit boards, International Journal of Production Research 33 (1995) 757–776.
[7] J.I. van Zante-de Fokkert, Process flow design for repetitive assembly, Ph. D. Thesis of the Technical University of Eindhoven, 1998.
[8] J. Ahmadi, S. Grotzinger, D. Johnson, Component allocation and partitioning for a dual delivery placement machine, Operations Research 36 (1988) 176–191.
[9] R.G. Askin, M. Dror, A.J. Vakharia, Printed circuit board family grouping and component allocation for a multi-machine, open shop assembly cell, Naval Research Logistics 41 (1994) 587–608.
[10] Z. Drezner, S. Nof, On optimizing bin picking and insertion plans for assembly robots, IIE Transactions 16 (1984) 262–270.
[11] R.A. Walas, R.G. Askin, An algorithm for NC turret punch press tool location and hit sequencing, IIE Transactions 16 (1984) 280–287.
[12] T. Leipälä, O. Nevalainen, Optimization of the movements of a component placement machine, European Journal of Operational Research 38 (1989) 167–177.
[13] K. Broad, A. Mason, M. Rönnqvist, M. Frater, Optimal robotic component placement, Journal of the Operational Research Society 47 (1996) 1343–1354.
[14] L.K. Moyer, S.M. Gupta, Simultaneous component sequencing and feeder assignment for high speed chip shooter machines, Journal of Electronics Manufacturing 6 (1996) 271–305.
[15] P.J. Egbelu, C. Wu, R. Pilgaonkar, Robotic assembly of printed circuit board with component feeder location consideration, Production Planning and Control 7 (1996) 162–175.
[16] J.H. Ahmadi, R. Ahmadi, H. Matsuo, D. Tirupati, Component fixture positioning for printed circuit board assembly with concurrent operations, Operations Research 43 (1995) 444–457.
[17] L.K. Moyer, S.M. Gupta, SMT feeder slot assignment for predetermined component placement paths, Journal of Electronics Manufacturing 6 (1996) 173–192.
[18] T. Horak, R.L. Francis, Utilization of machine characteristics in PC board assembly, Working Paper, University of Florida, Gainesville, FL, 1995.
[19] J.F. Bard, R.W. Clayton, T.A. Feo, Machine setup and component placement in printed circuit board assembly, the International Journal of Flexible Manufacturing Systems 6 (1994) 5–31.
[20] Y. Crama, A.W.J. Kolen, A.G. Oerlemans, F.C.R. Spieksma, Throughput rate optimization in the automated assembly of printed circuit boards, Annals of Operations Research 26 (1990) 455–480.
[21] P.J.M. van Laarhoven, W.H.M. Zijm, Production preparation and numerical control in PCB assembly, International Journal of Flexible Manufacturing Systems 5 (1993) 187–207.
[22] R.L. Francis, H.W. Hamacher, C.-Y. Lee, S. Yeralan, Finding placement sequences and bin locations for cartesian robots, IIE Transactions 26 (1994) 47–59.
[23] L.R. Foulds, H.W. Hamacher, Optimal bin location and sequencing in printed circuit board assembly, European Journal of Operational Research 66 (1993) 279–290.
[24] T.A. Younis, T.M. Cavalier, On locating part bins in a constrained layout area for an automated assembly process, Computers and Industrial Engineering 18 (1990) 111–118.
[25] I. Viczián, Finding placement sequences and bin locations for cartesian robots, Working paper, University of Würzburg, 1993.
[26] L.F. McGinnis, J.C. Ammons, M. Carlyle, L. Cranmer, G.W. Depuy, K.P. Ellis, C.A. Tovey, H. Xu, Automated process planning for printed circuit card assembly, IIE Transactions 24 (1992) 18–30.

[27] A. Dikos, P.C. Nelson, T.M. Tirpak, W. Wang, Optimization of high-mix printed circuit card assembly using genetic algorithms, Annals of Operations Research 75 (1997) 303–324.

[28] M. Gronalt, M. Grunow, H.O. Günther, R. Zeller, A heuristic for component switching on SMT placement machines, International Journal of Production Economics 53 (1997) 181–190.

[29] Y. Crama, O.E. Flippo, J.J. van de Klundert, F.C.R. Spieksma, The assembly of printed circuit boards: A case with multiple machines and multiple board types, European Journal of Operational Research 98 (1997) 457–472.

[30] T.F. Carmon, O.Z. Maimon, E.M. Dar-el, Group set-up for printed circuit board assembly, International Journal of Production Research 27 (1989) 1795–1810.

[31] A. Balakrishnan, F. Vanderbeck, A tactical planning model for mixed-model electronics assembly operations, CORE Discussion paper 9349, Catholic University of Louvain, Louvain-la-Neuve, 1993.

[32] S. Jain, M.E. Johnson, F. Safai, Implementing setup optimization on the shop floor, Operations Research 44 (1996) 843–851.

[33] M.J. Rosenblatt, H.L. Lee, The effects of work in process inventory costs on the design and scheduling of assembly lines with low throughput and high component costs, IIE Transactions 28 (1996) 405–414.

[34] E. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, The Traveling Salesman Problem: A Guided Tour, Wiley, New York, 1985.