On the Work Process Organization of Surface Mounted Component Printing

Jouni Smed
Mika Johnsson
Mikko Puranen
Jouni Lehtinen
Timo Leipälä
Olli Nevalainen

Turku Centre for Computer Science
TUCS Technical Report No 139
1997
ISBN 952-12-0092-8
ISSN 1239-1891

Abstract

The arrangement of operations in a production line for mounting the surface components on a printed circuit board is discussed. The production environment has a wide range of different products, which causes frequent set-up operations. The overall productivity of the line depends heavily on how the printing operations are organized. Set-ups cause delays which can be cut down by selecting carefully the feeders for the components and by solving a suitable sequence for the products. Algorithms which solve both of these problems are presented along with a discussion of the mathematical 0/1 integer programming approach. The revision of the production planning system has had a major impact on productivity. An increase of ca. 58 percent in the number of component insertions per hour is observed.

Keywords: flexible manufacturing systems, printed circuit boards, surface mounting, mathematical optimization, approximative algorithms

TUCS Research Group
Algorithmics

1 Introduction

In flexible manufacturing system (FMS) [9] several different types of products are manufactured by the same production facilities. FMS reduces the machinery investments and widens the product range. However, it also provides us with many challenging production planning and management problems.

The production program comprises batches of different products, each demanding special set-ups, parts, tools, and numerically controlled execution (NCX) codes. We suppose in this paper that the set-up times caused by the product changes are non-negligible, and therefore we can decrease the *set-up costs* by sequencing the products suitably.

In addition to the goal of minimizing the set-up costs, the production must meet *due dates*. The management wants to utilize the production facilities maximally and to cope with the short delivery times. At the same time it is preferable if the finished products do not pile up in storages. These two requirements—reliability and optimized usage of machinery—form the basis of the *just-in-time* (JIT) operation principle.

We can control the finishing times—and in this way also the fulfillment of the due dates—by *scheduling* the jobs. It should be noted that in many cases the production planner is mainly concerned with the due dates, and the minimization of the set-up times may sometimes conflict with this goal. See [3] for a problem where the aim is to minimize the number of late jobs. The construction of a feasible and, preferably, an efficient schedule is one of the most difficult tasks in production planning. In practical cases the problem is too complicated to be solved accurately (even in theory) [2, 5]. Therefore, the problem is usually approached by the use of approximative algorithms.

In this paper we present a real world situation where we can concentrate on a single machine despite the fact that the machine is only a part of the actual production process. The reason for this simplification is the dominating role of this work phase compared to all other phases in the system. To be more specific, our research originates from an existing *printed circuit board* (PCB) assembly plant (Teleste Corporation, Nousiainen, Finland) and aims at a solution for everyday use. The production line considered here is capable of manufacturing a very large selection of different *jobs* (i.e., batches of PCBs). During the last two years the number of different PCB types has been over 300, and their respective annual production volumes vary from one to several thousand boards. Therefore, the production is highly flexible and demands several set-ups daily.

This paper is divided as follows. We start in section 2 by describing the component printing line and the inefficiencies in its operation. This section contains some technical details which are needed in order to clarify the aspects of the problem in question. Our proposal for a revised production planning system is described in section 3. In section 4 we describe a new method for updating the control programs of the machine. We study how the component feeders should be filled in section 5, and what are the possibilities to improve the overall production by grouping the products in section 6. The system is described in section 7. An evaluation of the effects of the new system is given in section 8. Concluding remarks appear in section 9.

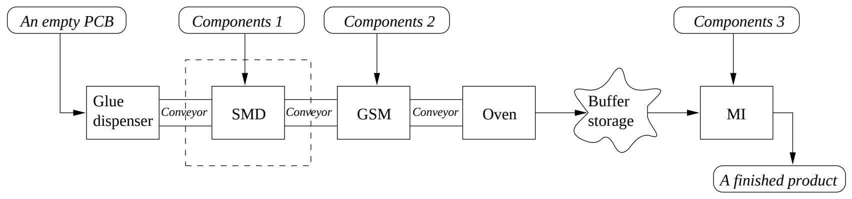

Figure 1: The production phases of the surface mounted onsertion. Legend: SMD = surface mounting device, GSM = general surface mounting device, MI = manual insertion

2 The Production Environment

Let us consider a typical assembly line for automatic component printing, see figure 1. The line comprises several successive work phases. An initially empty PCB passes first a *glue dispenser* which inserts a glue dot at each onsertion locus or draws adhesive paste over the whole board in order to fixate the electric components. The actual printing is done by two onsertion machines. The first one, *surface mounting device* (SMD), is adapted to fast operation and it is used for the majority of the component onsertions. Components which require specialized tools are onserted by a more flexible but slower robot, a *general surface mounting* (GSM) machine. These two onsertion phases are followed by an *oven* which heats the PCBs in order to harden the glue/paste. After that the PCBs wait in a *buffer storage* and finally pass a *manual insertion* phase in which some large components are inserted and soldered.

If we look at the different jobs which are processed on the line, we notice that their number is very high and the amount of PCBs in a job is usually small. The daily production program includes typically 4–10 different *products*. The set-up times form a significant part of the total production time—it can be as much as 50 percent. Therefore, our main objective is to minimize the set-up times by arranging the products efficiently. Normally the due dates are considered the most important restriction, but in this case they are managed by a two-level priority classification: products are either *urgent* or *non-urgent*. The last feature that affects the overall production time is that there are two different widths for the PCBs. Therefore, if the next PCB has a different width than the previous one, the width of the conveyor must be changed.

There are many reasons why the SMD machine [14] is the bottleneck of the whole production line. Its set-ups and component printing consume most of the production time. We do not consider the balance between the SMD and GSM machines here because the frequent product changes make this kind of balancing hard to accomplish, although they are capable of printing the same components.

The difficult management of the machine in a *multi-model production* environment emanates from the flexibility of the SMD machine. It gets the surface mounted components from six *carriage modules*, see figure 2. The following technical details should be taken into consideration:

1. Each carriage includes 80 linearly arranged *feeder slots* which contain the components. A component takes at least two slots giving thus each car-

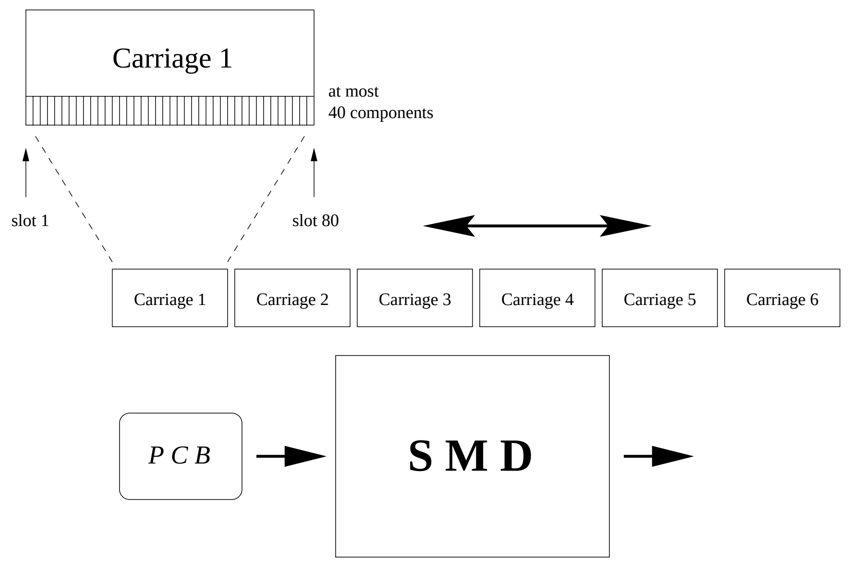

Figure 2: A schematic view of the SMD printing machine

riage a theoretical maximum capacity of 40 components, and 240 components for the whole machine. Because the sum of different component types in a PCB is significantly smaller than the capacity of the machine, we can quite freely choose an appropriate input organization.

2. The two outermost carriages can be separated from the unit while the machine is operating. The operator can carry out the set-up procedure for carriages 1 and 2 (or 5 and 6) while carriages 3–6 (or 1–4) are active in the component printing. While the machine is working, the set-up of the midmost carriages 3 and 4 cannot be changed. Therefore, they should contain the most commonly used components.

Ammons *et al.* [1] divide *component set-ups* into two categories: the *standard set-up* comprises components that are staged for every PCB while the remaining components are defined in the *custom set-up*. Previously carriages 2–4 were filled with the standard set-up components. The board was then printed with components from carriages 1–4 if the required components were contained by the standard set-up carriages 2–4 and the rest could be set up in carriage 1. Otherwise, the board was printed with standard set-up components from carriages 3–4 and custom set-up components from carriages 5–6.

The above-described method was, however, inefficient because of the following reasons:

1. The printing programs were laborious to update, which caused that the standard set-up components were seldomly changed. Therefore, the standard set-up gradually corrupted to contain components whose demand was not maximal any more. In addition, there was no efficient method for solving a new standard set-up.

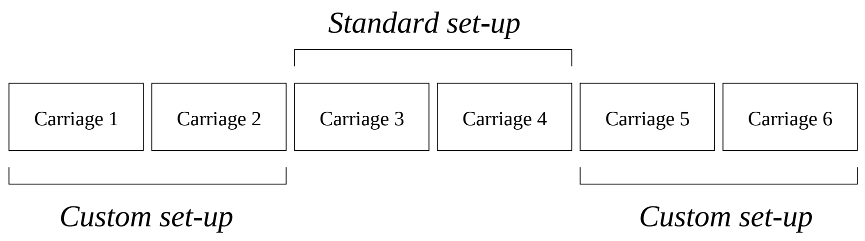

Figure 3: An improved partition of the carriages

2. Each product required a separate set-up because each printing program required a new setting for the custom set-up components. Furthermore, it was seldom possible to print one product during the set-up of another.

3. The order of the components in the feeders was not very efficient, which further reduced the productivity.

4. The printing order was poor.

3 The Structure of the New System

The revised production planning system solves the inefficiences described above by introducing the following new features:

1. a method for choosing the standard set-up (section 5),

2. forming groups of jobs with the same feeder setting (section 6),

3. feeder optimization for product groups, and

4. using an efficient code generator that utilizes the current feeder set-up.

However, these features require a flexible and dynamic update process for the printing programs which is described in section 4. For the feeder and printing order optimization we use previously developed methods [7].

The new system uses a different partition of standard and custom set-up carriages, see figure 3. Carriages 3 and 4 contain now a standard set of components, thus leaving on both ends symmetrically two carriages for the custom set-up. This layout enables continuous printing: while PCBs are being printed, the components of the next set-up can be placed on carriages which, at the time, remain idle on the other end of the machine. There are three different strategies for the feeder set-up [1]:

1. a static set-up which is identical for all products,

2. a set-up which is identical for only a part of the products (family set-up), and

3. every product has its own set-up.

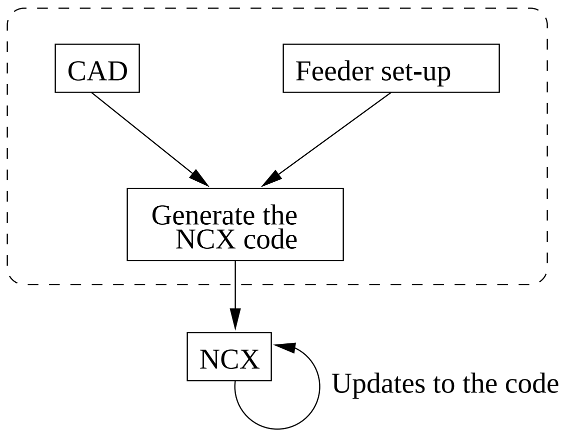

Figure 4: Updating the printing programs in the old system

The first strategy is straightforward but, unfortunately, it is not applicable in our case because the total number of different component types exceeds the feeder capacity. In addition, the static feeder set-up excludes the possibility to improve the printing speed. The third strategy resembles the situation before the introduction of the system presented in this paper.

The second strategy is a compromise: here we form standard set-ups for product groups (i.e., different products can use the same set-up). This approach is connected with the idea of (*product*) *families* (or *groups*) [8, 11]. Note that in the first strategy we have one single family whereas in the third strategy each product forms a singular family (or product group). The difficulty with the family set-up strategy is that once the families have been assigned, the jobs should be sequenced on family basis, not independently. Nevertheless, this strategy forms the basis of the solution presented in this paper: the products are divided into groups which are then sequenced by the production engineer.

4 Updating the Printing Programs

In the old system the printing programs were updated as depicted in figure 4. The NCX files (a file type used by Universal and Sanyo SMD printing machines) contain the feeder set-ups for each PCB. The program which creates them has two inputs: a CAD file of the PCB and a file containing the standard feeder set-up.

Previously, when there were changes in the PCB or updates to the printing data, the changes were made directly to the NCX file and not to the CAD file. If the changes were made to the CAD file, the NCX file would have to be recreated by using the CIMBrigde-system [12], which takes a long time and is generally a laborious task. The minor changes made to the file continued to add up until it was no longer efficient to use the original source files.

Figure 5 presents a solution to the problem: The NCX files are now updated by using a new program which uses the old NCX file and a new feeder set-up file as an input and updates the NCX file accordingly. The updated NCX file

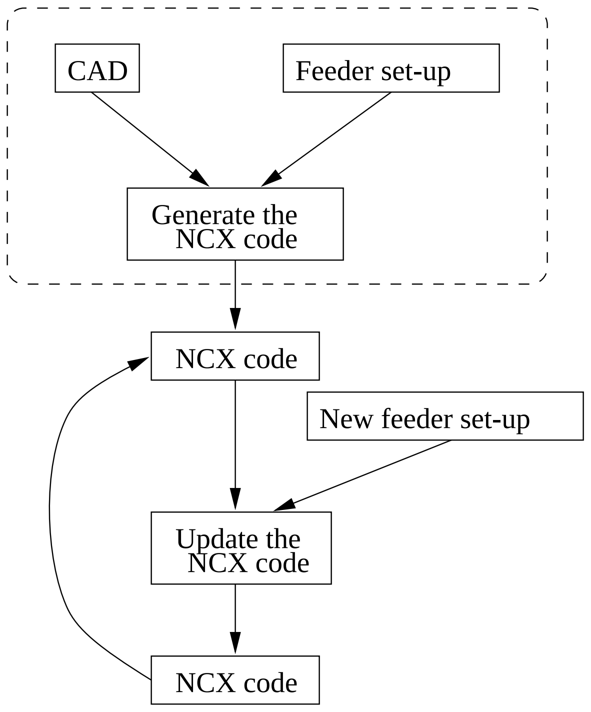

Figure 5: The revised update procedure

still contains the changes made to the original NCX file. The revised update procedure enables us to introduce new feeder set-ups, which was a troublesome operation in the old system. This is required when we introduce the concept of product groups in section 6. These groups are dynamic, and therefore every time a product is manufactured, it requires an updated NCX file.

5 Choosing the Standard Set-up

By using the production data we have to decide which components should belong to the standard set-up (i.e., in carriages 3 and 4). We could choose

- the most frequent components,
- the most frequent components from the PCBs containing the greatest number of components,
- the most frequent components from the PCBs containing the smallest number of components, or
- components from the most frequently manufactured PCBs.

The first three methods are based on the data from all the PCBs. However, the production focuses on certain PCBs. The last method isolates the most frequently manufactured PCBs in the current production and assigns a set-up which is based on the real component demand.

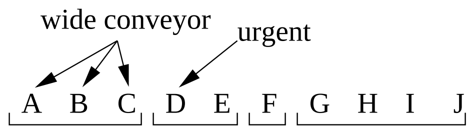

Figure 6: Grouping the jobs; a sample case

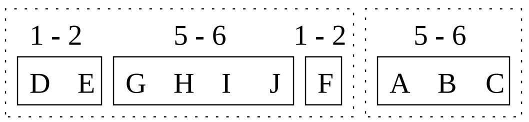

Figure 7: A sequence of the groups and carriage allocations

In the current system we pick first the most frequently produced PCBs. From these PCBs we select 80 most frequently used components (which take the available 160 feeder slots), omitting wide components and components which are slow to print. On the whole, choosing a standard set-up has turned out to be a noncritical factor with respect to the overall operation of the plant.

6 Grouping

In this section we return to the second feature of the planning system: forming product groups with identical feeder set-up. In section 3 we presented a strategy in which a part of the products has the same set-up. These products form a group [8, 11] and they are printed successively. In addition, there is no need for set-up operations between jobs belonging to the same group. If the printing of a group takes a longer time than the set-up of the next group, the group changing does not cause a delay and the PCBs can be printed continuously. Therefore, the products are classified into groups according to their *closeness* (i.e., the amount of mutual components). Products belonging to the same group will succeed each other (if possible). Thus, they use the same carriages and leave the other end of the carriage unit free for the next set-up.

Let us give an example of the grouping. Suppose we want to process jobs A, B,. . . , J (figure 6). Jobs ABC belong to the same group and use a wide conveyor, while the narrow PCBs form groups DE, F and GHIJ. Job D is urgent. Figure 7 shows a possible sequence for the groups. The group DE is processed first due to the urgency of job D. The components belonging to the custom set-up of the group DE are located in carriages 1 and 2. The custom set-up of the group GHIJ uses carriages 5 and 6, the next group (F) again in carriages 1 and 2 and the last group (ABC) in carriages 5 and 6. See figure 8 for an illustrative Gantt diagram of the processing.

We can use two grouping approaches:

1. In the *maximal grouping* we create as large groups as possible, i.e., we try to utilize the capacity of all four carriages. This minimizes the number of set-ups, but it has a negative effect on the printing times because the components are scattered in a large area in the carriages.

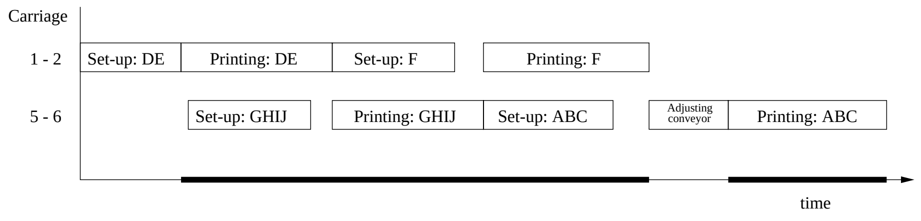

Figure 8: A Gantt diagram of the operations

2. In the *minimal grouping* we create smaller groups but keep them large enough to overlap the set-ups.

It is hard to tell which of the two approaches is preferable. The answer depends on the other demands of the line and its personnel. Here we should consider the job processing times and the component changing times. The latter is ca. 45 seconds for a component already in a reel, and ca. 120–180 seconds when the reel has to be loaded, too. The personnel, however, prefers the maximal grouping.

We introduce eight algorithms for grouping the jobs. The six algorithms presented in section 6.1 are greedy, and the algorithm in section 6.2 is optimal for making one maximal group. The eigth algorithm is introduced in section 6.3 and it is based on clustering techniques. The algorithms in section 6.1 and 6.2 use the same basic control flow:

```
while the group of the jobs is non-empty
    • determine a new job group
    • output the group
    • delete the jobs of the group
```

We study two aspects of the algorithms. First, we compare the heuristic solutions to the optimum solution in the case when the criterion is to form one maximal group (section 6.4). Secondly, we concentrate on how well the heuristics are able to minimize the number of groups (section 6.5). We have generated a set of random sample problems drawn from the whole product range (341 different products) where the problem size varies from 10 to 341. We have generated several parallel sample problems for sizes 10–40 in order to estimate the power of the algorithms for cases that occur frequently in production.

6.1 Heuristic Grouping

The algorithms differ in how they construct the next group. On a coarse level the methods work as follows:

Single Inclusion Method Add a new job to the current group of selected jobs so that its inclusion causes a minimal increase in the number of the different components in the group. Repeat the same procedure unless the capacity of the feeders is exceeded.

Double Inclusion Method The method is the same as in Single Inclusion but the jobs are added as pairs.

Triple Inclusion The method is the same as in Single Inclusion but the jobs are added as triplets.

Weighting Method The jobs are added in decreasing order of a weighted *similarity measure*

$$s_j = \frac{1}{p+1} \sum_{v=0}^{p} v N_v .$$

Here s_j is the similarity measure between the job j and the other jobs; p is the number of different component types on PCB j, and N_v is the number of those component types on PCB j that appear exactly on v other PCBs. For a PCB that contains a totally different set of components from the other PCBs, the similarity measure is zero, and for PCBs with very similar components, the measure is high.

Exclusion Method Begin with a group containing all PCBs. Exclude the PCB which releases the greatest number of feeder slots. Repeat the exclusion step until the components of the group fit in the carriages.

Best-of-five Method Choose the best method from the ones described above.

6.2 Mathematical Formulation for Grouping

We can formulate the product grouping problem as a mathematical optimization problem [13]. In this section we consider the stepwise construction of maximal groupings and give a formulation of this problem. For this model we introduce the following parameters:

d_i	the number of feeder slots needed by component i
m :	the number of different PCBs
n :	the number of different components
k :	the capacity of the carriage
a_{ij} :	the number of component i needed by board j

We denote the decision variables

$$x_i = \begin{cases} 1, & \text{if component } i \text{ is placed in the carriage} \\ 0, & \text{otherwise} \end{cases}$$

and

$$y_j = \begin{cases} 1, & \text{if board } j \text{ can be printed} \\ 0, & \text{otherwise.} \end{cases}$$

PCB j cannot be printed if there exists such a component i that $x_i = 0$ and $\min\{1, a_{ij}\} = 1$. Thus,

$$x_i = 0, \min\{1, a_{ij}\} = 1 \Longrightarrow y_j = 0,$$

where

$$\min\{1, a_{ij}\} = \begin{cases} 1, & \text{if board } j \text{ contains component } i \\ 0, & \text{otherwise.} \end{cases}$$

The three remaining value combinations of x_i and $\min\{1, n_{ij}\}$ do not specify y_i so that we obtain the restriction

$$y_j \leq 1 + x_i - \min\{1, a_{ij}\}.$$

If we want to maximize the number of the different PCBs which we can produce, we obtain the 0/1-optimization problem

$$\begin{cases} y_1 + y_2 + \cdots + y_m = \max! \\ d_1x_1 + d_2x_2 + \cdots d_nx_n \leq k \\ y_j - x_i \leq 1 - \min\{1, a_{ij}\}, i = 1, \ldots, n; j = 1, \ldots, m \\ x_i \in \{0, 1\}, i = 1, \ldots, n; y_j \in \{0, 1\}, j = 1, \ldots, m \end{cases}$$

with $n+m$ variables and $nm+1$ restrictions. We can also maximize the weighted sum $\sum c_j y_j$ of the number of PCBs where the weight could be related to the production quantities. If we for some reason want component l to reside in the carriage, we might add the constraint $x_l = 1$.

In the above formulations restrictions are not needed in the case $\min\{1, a_{ij}\} = 0$ when component i is not used in board j. Then we have a problem

$$\begin{cases} y_1 + y_2 + \cdots + y_m = \max! \\ d_1x_1 + d_2x_2 + \cdots d_nx_n \leq k \\ y_j - x_i \leq 0, \forall i, j \ni \min\{1, a_{ij}\} = 1 \\ x_i \in \{0, 1\}, i = 1, \ldots, n; y_j \in \{0, 1\}, j = 1, \ldots, m \end{cases}$$

with only $1 + \sum\sum \min\{1, a_{ij}\}$ restrictions. Note that the above problem formulation searches only for one optimal grouping and leaves the other products unselected. When the procedure is repeated, we obtain a speedy, locally optimal heuristic. We call it Stepwise Optimal method.

6.3 Clustering Method

This method is related to the clustering techniques used in other contexts, cf. PNN algorithm [4]. Among the several different variations we consider the following method: Begin with forming singular groups so that each PCB forms a group of its own. Search for group pairs that can be merged into a single group which does not exceed the feeder capacity. Select the pair which gives the greatest benefit among all the possible pairs (i.e., if the set R_i contains the components of the group i, and R_j the components of the group j, the pair (i, j) is selected so that $|R_i| + |R_j| - |R_i \cup R_j|$ is maximal). Merge the groups and repeat the merging procedure if possible.

6.4 Maximal Group Size

We can conclude from the running time curves of the optimal method (see figure 9) that heuristic algorithms are needed in order to solve the problem in a reasonable time. The growth of the running time for the heuristic algorithms

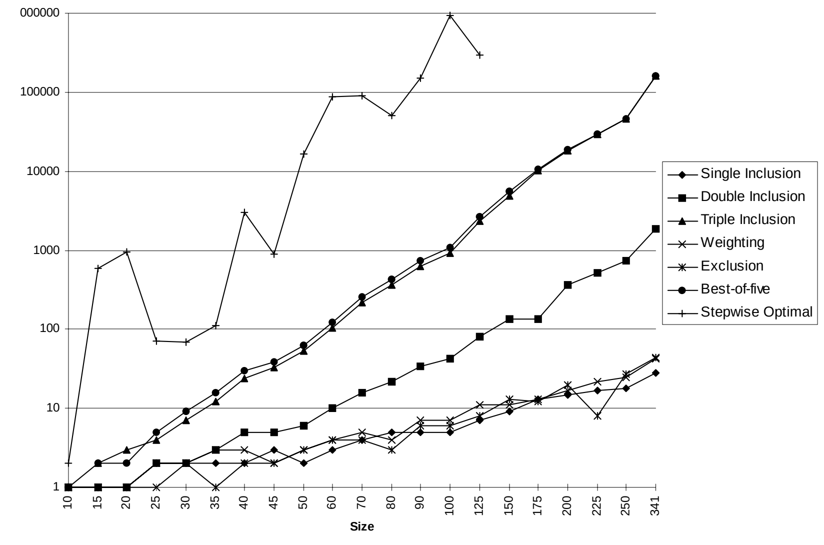

Figure 9: Running times on a Pentium 133 MHz computer (note the logarithmic scaling)

is not completely linear because the problems have been generated randomly. Nevertheless, they give us some indication whether the method is feasible or not. We use a 0/1-programming system (lp-solve [10]) for solving the mathematical optimization problem.

We note that the Triple Inclusion heuristic consumes significantly more time than the other heuristics, and therefore it dominates the running time of the Best-of-five method, too. In addition to the optimal method, the Triple Inclusion must be ruled out when the size of the problem is comparatively large. The Double Inclusion method turns out to have a shorter running time and it gives good results (98.5 percent of the optimal on the average, see table 2). This reflects the benefits of choosing two jobs at a time instead of one (as in the Single Inclusion method which gives 95.5 percent of the optimal) when we form a group. However, the additional improvement gained by taking three jobs (the Triple Inclusion performs only 0.24 percent better than the Double Inclusion) is outweighted by the long running time. Therefore, we did not consider variants which use four or more jobs at a time. Much to our surprise, the performance of the Exclusion method is quite mediocre. In this case when the optimization criterion is to form one maximal group, the Weighting method performs poorly. However, it takes into account other aspects of the problem, which is shown more clearly in the next section.

Table 1 shows the size of the maximal group for different problem sizes. We have used this knowledge and calculated in table 2 the average ratio in comparison to the optimum size.

Problem size	Single Inclu-sion	Double Inclu-sion	Triple Inclu-sion	Weight-ing	Exclu-sion	Best-of-five	Step-wise Optimal
10	10	10	10	10	10	10	10
15	9	10	9	7	10	10	10
20	11	11	11	7	11	11	11
25	15	16	15	10	16	16	16
30	17	18	18	11	14	18	18
35	25	25	25	16	25	25	25
40	19	20	21	9	15	21	21
45	26	27	28	21	26	28	28
50	22	24	24	14	19	24	24
60	33	33	33	17	26	33	33
70	33	34	34	12	23	34	35
80	35	36	36	25	24	36	37
90	36	37	38	24	23	38	38
100	44	45	45	30	29	45	45
125	54	55	55	21	41	55	56
150	57	57	58	27	50	58	N/A
175	64	65	65	38	52	65	N/A
200	66	67	70	45	69	70	N/A
225	61	61	63	35	66	63	N/A
250	82	85	88	41	75	88	N/A
341	93	94	102	44	91	102	N/A

Table 1: The maximum size of the first group for different problem sizes and solution methods

Algorithm	%
Single Inclusion	95.58
Double Inclusion	98.53
Triple Inclusion	98.77
Weighting	57.49
Exclusion	76.66
Best-of-five	99.26

Table 2: Comparing the heuristics to the optimum

Size	Single Inclusion	Double Inclusion	Triple Inclusion	Weighting	Exclusion	Best-of-five	Stepwise Optimal	Clustering
10	1	1	1	1	1	1	1	1
15	3	3	3	3	3	3	3	3
20	3	3	3	3	3	3	3	3
25	4	4	4	4	4	4	4	4
30	4	4	4	4	5	4	4	4
35	4	4	4	5	4	4	5	4
40	6	6	6	6	6	6	6	6
45	6	6	5	4	5	5	5	5
50	6	6	6	6	7	6	6	5
60	8	8	7	7	8	7	7	7
70	8	9	9	9	9	9	8	8
80	8	8	7	7	9	7	7	7
90	10	10	10	11	10	10	9	9
100	10	10	10	11	10	10	10	9
125	12	13	12	14	13	12	11	11
150	13	13	13	14	14	13	N/A	11
175	13	14	13	16	14	13	N/A	13
200	15	16	16	18	16	16	N/A	15
225	17	17	16	20	17	16	N/A	14
250	18	17	16	22	22	17	N/A	17
341	21	20	20	28	25	20	N/A	19

Table 3: The number of different groups produced by the solution methods (N.B., the results of the optimal method are not available from size 150 onwards because of the immense running time)

6.5 Minimizing the Number of Groups

We will now examine how well different methods are able to minimize the number of different groups and thereby also the number of set-ups. The Stepwise Optimal method, which forms maximal groups one at a time, does not ensure that we get a minimal number of different groups every time. In the previous section we noticed that the Weighting method gives poor results when we try to form one maximal group (on the average only 57.5 of the optimum size). However, the Weighting method can sometimes form less groups than the Stepwise Optimal method (e.g., for problem size 45 in table 3).

The number of different groups given by each method is shown in table 3. We notice that the Clustering method provides us with the best results—especially when the size of the problem is quite large (see table 4). Its running time is similar to the Single Inclusion method. It is also notable that even if different methods give the same number of groups, the distribution of the jobs between the groups can vary greatly (see figure 10). The Weighting and Clustering methods tend to form (more or less) equally sized groups while the other methods usually form one large and several small groups.

The results of the other heuristics are compared to the best method (Clus-

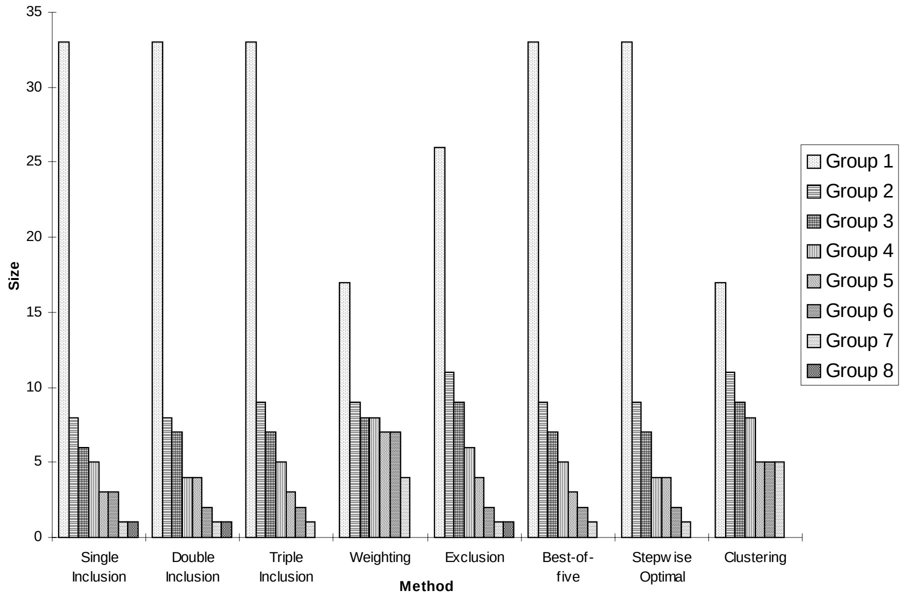

Figure 10: Group size histograms for a sample case of 60 products

Algorithm	%
Single Inclusion	94.7
Double Inclusion	93.8
Triple Inclusion	97.3
Weighting	84.5
Exclusion	87.8
Best-of-five	96.8

Table 4: Comparing the heuristics to the Clustering Method

tering) in table 4. The ratios confirm that the Weighting method is somewhat weak but its value lies in its ability to produce an even distribution of the group sizes. The performances of the Inclusion methods and Best-of-five are on about the same level. Because the demands in the production can change (i.e., the production engineer may prefer to have one large group or several equally sized groups), we have included the Single Inclusion, Weighting, Exclusion and Clustering methods in our production planning system. This way the production engineer gets in a reasonable time different groupings of the production plan to choose from.

7 System Description

The system depicted here includes tools for grouping the jobs of the production plan and optimizing the set-up for each group [6]. When using the system, the production engineer can

- choose the products from a product list,
- choose which method (Single Inclusion, Weighting, Exclusion or Clustering) is applied to form the groups,
- assign the standard set-up,
- assign the custom set-up for a given group and carriers (1–2 or 5–6),
- produce optimized NCX codes for a whole group or for some jobs within a group,
- compare two groups in order to discern mutual components and their respective locations,
- view the current groups by listing the jobs, the components or the feeder set-up,
- remove groups,
- inspect whether a new job can be inserted to some existing group, and
- edit the component library.

In the current system the production engineer can freely re-sequence the groups and the jobs within a group. The latter does not affect the set-up times since no set-up is needed for jobs in the same group. The priority of the job, the batch size, the conveyor width and the availability of required components are the most important factors when constructing a feasible schedule.

8 Experiences

In order to demonstrate the performance of the new system we present an actual production plan for a period of one week, see table 5. The production schedule is created by using both the old system and the new system. We can calculate the number of required component changes, component change operations (i.e.,

PCB	pcs.	Old	New	Improvement	%	Group
D2284A1	200	48.61	32.64	15.97	32.9	1
CRT212	30	96.62	70.90	25.71	26.6	1
D5250A	100	55.06	40.73	14.34	26.0	1
D5450B1	210	42.87	34.13	8.75	20.4	2
CRR212C	35	103.16	83.70	19.47	18.9	1
D2187B1	31	33.15	27.05	6.09	18.4	1
D2161A1	79	31.77	26.21	5.56	17.5	1
D2481A	100	37.42	30.92	6.50	17.4	1
D2281A	200	38.95	32.45	6.50	16.7	1
DXA821RE	52	37.52	31.78	5.74	15.3	1
S2203	30	47.46	41.20	6.26	13.2	1
M3151EC2	30	105.90	93.49	12.41	11.7	3
AXP2011	50	39.19	35.10	4.09	10.4	2
MHE6107	47	78.28	70.47	7.80	10.0	2
DXO802	200	37.10	33.48	3.62	9.8	1
A80430B1	30	62.06	57.42	4.64	7.5	2
DTU082B1	31	30.78	28.73	2.06	6.7	1
AXF246D	30	22.52	21.96	0.56	2.5	1
AXA860	56	36.73	36.73	0.00	0.0	1
M3153CC2	10	104.30	104.53	−0.24	−0.2	1
DTU121B1	31	27.63	29.70	−2.07	−7.5	1
Total		1,117.07	963.31	153.76	13.8	

Table 5: An analysis of a sample schedule for a period of one week. Legend: Old = production time in the old system (min), New = production time in the revised system, Improvement = difference between Old and New, % = difference in percents, Group = group to which the product belongs

set-ups) and the expected printing time for each board by using a simulator presented in [7]. From this data we can calculate the total time required to produce the whole production program in both systems.

The benefits of the new system are obvious (see table 6). In the old system we have to do 21 set-ups, one for each product, while in the new system products are divided into three groups, each requiring a single set-up. Also, the printing time is vastly reduced; in this case the reduction is ca. 18 percent. It must be noted that we have included only the times which differ in the new and the old system. However, there is a lot of processing (program loading, line changing, coping with breakdowns and other problems) which is not affected by the change of the system, and therefore this analysis is not complete. We believe that the decreased number of set-ups will increase the production much more than these estimates suggest, the reason being that when there are only few set-ups, the whole organization of other tasks works more efficiently because of less interruptions.

The new system has been in production use since May 1997. The manufacturer has collected statistical data from the production. If we compare the first ten-week period after the change with the preceding twenty-week period, we observe the following improvements:

	Number of set-ups	Number of feeder changes	Total set-up time	Total onser-tion time	Total produc-tion time
Old	21	294	30,240 s	72,868 s	103,108 s
New	3	246	16,560 s	60,067 s	76,627 s
Improvement	18	48	13,680 s	12,801 s	26,481 s
%	85.7	16.3	45.2	17.6	25.7

Table 6: Comparing the old and new system

- The average number of component placements per hour (of total time) has increased by 57.6 percent.
- The average number of component placements per hour (of the actual printing time) has increased by 16 percent which is in accord with the results of table 5. The main reason for this is the improvements in printing and feeder optimization (cf. [7]).
- The average number of completed jobs in a week has increased from 22 to 28.
- The average time to change from one job to another (including machine set-up and other delays) on the whole line has decreased from an hour and a half to one hour (35.5 percent).

We realize that there has been minor changes in the product assortment during the test period. The average size of a batch has slightly increased which causes that the number of placements per hour (of total time) given here is somewhat optimistic. On the other hand, the number of completed jobs per week suffers from the growing batch sizes. Also, a slight improvement is due to purchasing new feeder reels which speeds up the set-up.

9 Concluding Remarks

We discussed the production planning in a flexible manufacturing line. The situation was abstracted as the management of a single machine. Even with this simplification we were confronted by a number of hard optimization problems. We solved the grouping problem and the code generation problem separately. These solutions formed the basis for the new system which introduced new ways to solve the scheduling problems into the production plant. The system described in this paper is in daily use and has decreased the change time of a job by 35 percent. The number of component placements per hour has increased by 58 percent due to the new system and code optimization.

There remains yet a number of open research questions we would like to solve, for example, how to sequence the groups efficiently. Furthermore, the grouping of the jobs is connected to the due dates, the board widths and production volumes, which were bypassed in this paper. The division between the SMD and GSM machines should be reconsidered in order to gain a better balance for the load of the whole production line.

We presented algorithms which group the products heuristically. Because there are several conflicting goals when forming the groups, we are currently developing new algorithms which are based on the fuzzy multiple criteria optimization. This approach may have some potential in improving the standard set-up, too.

In the next version of the system the scheduling of the groups is partly automatized (only partly, because the production engineer likes to preserve the flexibility of the current system).

References

[1] Ammons, J. C., Carlyle, M., Cranmer, L., DePuy, G., Ellis, K., McGinnis, L. F., Tovey, C. A., Xu, H., "Component Allocation to Balance Workload in Printed Circuit Card Assembly Systems", *IIE Transactions*, (accepted for publication)

[2] Brucker, P., *Scheduling Algorithms*, Springer-Verlag, 1995

[3] Crauwels, H. A. J., Potts, C. N., Van Wassenhove, L. N., "Local Search Heuristics for Single-machine Scheduling with Batching to Minimize the Number of Late Jobs", *European Journal of Operational Research* 90, pp. 200–13, 1996

[4] Equitz, W. H., "A New Vector Quantization Clustering Algoritm", *IEEE Transactions on Acoustics, Speech and Signal Processing* 37/10, pp. 1568–75, 1989

[5] Garey, M. R., Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, 1979

[6] Johnsson, M., *User Manual for Grouping Software*, M-97-1, University of Turku, 1997

[7] Johnsson, M., Leipälä, T., Nevalainen, O., "Optimizing Turret Based SMD Machines", in preparation

[8] Kumar, K. R., Kusiak, A., Vanelli, A., "Grouping of Parts and Components in Flexible Manufacturing Systems", *European Journal of Operational Research* 24, pp. 387–97, 1986

[9] Kusiak, A., "Application of Operational Research Models and Techniques in Flexible Manufacturing Systems", *European Journal of Operational Research* 24, pp. 336–45, 1986

[10] lp-solve, system available: `ftp://ftp.ics.ele.tue.nl/pub/lp-solve/`

[11] Maimon, O., Shutb, A., "Grouping Methods for Printed Circuit Boards", *International Journal of Production Research* 29/7, pp. 1370–90, 1991

[12] Mitron Corporation, *Line Scheduler User Manual*, Version 3.2, 1995

[13] Nemhauser, G. L., Wolsey, L. A., *Integer and Combinatorial Optimization*, Wiley, 1988

[14] Universal Instruments Corporation, *Model 4792 Machine Operations Manual*, 2nd Edition, 1996