Annals of Operations Research 26(1990)455–480

THROUGHPUT RATE OPTIMIZATION IN THE AUTOMATED ASSEMBLY OF PRINTED CIRCUIT BOARDS

Y. CRAMA, A.W.J. KOLEN and A.G. OERLEMANS
Department of Quantitative Economics, University of Limburg, P.O. Box 616, NL-6200 MD Maastricht, The Netherlands

F.C.R. SPIEKSMA
Department of Mathematics, University of Limburg, P.O. Box 616, NL-6200 MD Maastricht, The Netherlands

Abstract

The electronics industry relies heavily on numerically controlled machines for the placement of electronic components on the surface of printed circuit boards (PCB). This paper proposes a heuristic hierarchical approach to the problem of optimizing the throughput rate of a line of several of such machines, devoted to the assembly of a single type of PCB. A number of well-known NP-hard problems emerge, for which mathematical models and heuristic solution methods are developed. The approach is tested on a real-life problem, for which it is shown to perform well.

Keywords

Hierarchical scheduling, heuristics, electronic industries, printed circuit board, flexible assembly systems.

1. Introduction

The electronic industry relies heavily on numerically controlled machines for the placement of electronic components on the surface of printed circuit boards (PCB). These placement (or mounting, or pick-and-place) machines automatically insert components into PCBs in a sequence determined by the input program. The most recent among them are characterized by high levels of accuracy and speed, but their throughput rates still appear to be extremely sensitive to the quality of the instructions. On the other hand, the effective programming of the machines becomes steadily more difficult in view of the increasing sophistication of the available technology. The development of optimization procedures allowing the efficient operation of such placement machines therefore provides an exciting challenge for the operations research community, as witnessed by, e.g., the recent papers by Ahmadi et al. [1], Ball and Magazine [2], and Leipälä and Nevalainen [14].

In this paper, we propose a hierarchical approach to the problem of optimizing the throughput rate of a line of several placement machines devoted to the assembly of a single product. As is usual in the study of flexible systems, the high complexity of the problem suggests its decomposition into more manageable subproblems, and accepting the solution of each subproblem as the starting point for the next one. Of course, this methodology cannot guarantee the global optimality of the final solution, even assuming that all subproblems are solved to optimality. This is even more true in the present case, where most subproblems themselves turn out to be NP-hard, and hence can only be approximately solved by heuristic procedures. Nevertheless, such hierarchical approaches have previously proved to deliver good quality solutions to similarly hard problems (e.g. in VLSI-design; see Korte [11]). They also offer the advantage of providing precise analytical models for the various facets of the global problem (see, for example, Buzacott and Yao [4] for a discussion of analytical models in FMS).

Our approach has been tested on some industrial problems, but more experimentation would be required in order to precisely assess the quality of its performance and its range of applicability. In particular, as pointed out by one of the referees, the validity of some of our models is conditioned by the validity of some exogenous assumptions about the nature of instances "coming up in practice" (see, for instance, section 4.1). Even though these assumptions were fulfilled in the industrial settings that motivated our study, they may well fail to be satisfied in other practical situations. This would then invalidate the use of the corresponding models. However, we believe that the hierarchical scheme and most of the techniques presented in this paper would nevertheless remain applicable for a wide range of problem instances.

We now give a brief outline of the paper. The next section contains a more detailed description of the technological environment, and section 3 provides a precise statement of the problem and a brief account of previous related work. Sections 4 and 5 present our approach to the solution of the throughput rate optimization problem. Section 4 addresses the workload balancing problem for the line of machines, and section 5 deals with the optimal sequencing of operations for individual machines. Both sections present mathematical models and heuristic solution methods for the various subproblems arising in our decomposition of the global problem. Finally, in section 6 we describe the results supplied by our approach on a practical problem instance.

2. Technological environment

In this paper, we are concerned with the automated assembly of a number of identical PCBs. For our purpose, the assembly of a PCB consists of the insertion of electronic components of prespecified *types* (indexed by $1, \ldots, T$) into prespecified *locations* (indexed by $1, \ldots, N$) on a board. Prior to operations, the components of different types are collected on different *feeders* (one type per feeder). Feeders are used by the placement machines as described below. We denote by N_t the number of components of type t ($t = 1, \ldots, T$). So, $N = \sum_{t=1}^{T} N_t$.

We assume that a line of M placement machines is devoted to the assembly of the PCBs. The machines we have in mind are of the CSM (Component Surface Mounting) family. They feature a *worktable*, a number S of *feeder slots*, and three *pick-and-place heads* (see fig. 1).

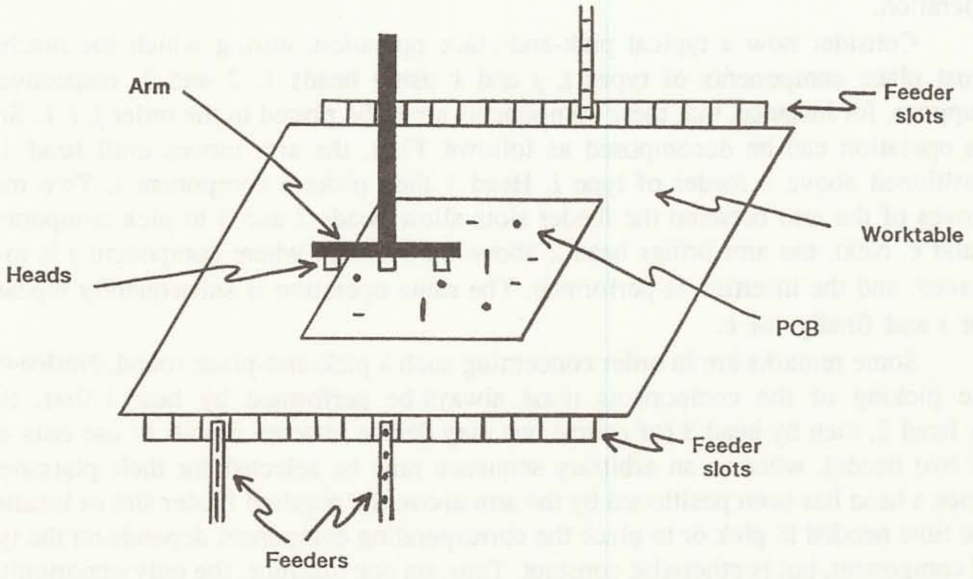

Fig. 1. Schematic representation of a placement machine.

The PCB is carried from one machine to the next by an automatic transport band until it comes to rest on the worktable. It stays motionless during the mounting operations.

The feeder slots are fixed to two opposite sides of the worktable, $S/2$ of them on each side. The feeders containing the components to be placed by the machine must be loaded in the slots before the mounting begins. Depending on its type, each feeder may require 1, 2, or even more adjacent slots.

The pick-and-place heads are numbered from 1 to $3M$, with heads $3m - 2$, $3m - 1$ and $3m$ on machine m (but, for short, we shall also refer to heads 1, 2 and 3 of each machine). They are fixed along the same *arm*, which always remains parallel to the side of the worktable supporting the feeder slots. The arm can move in a horizontal plane above the worktable. It can perform vertical moves to allow the heads to pick components from the feeders or to insert components into the board.

Each head can carry at most one component at a time. It must be equipped with certain tools (chucks and nozzles) before it can handle any components. The collection of tools necessary to process a given component we call *equipment*. With every component type is associated a restricted set of alternative equipment by which it can be handled. In most situations, four or five equipments suffice to mount all component types. Changing the equipment of a head can be done either manually or automatically, depending on the technology (notice that, on certain types of machines, an equipment change can be performed automatically for heads 1 and 2, but only manually for head 3). In either case, an equipment change is a time-consuming operation.

Consider now a typical pick-and-place operation, during which the machine must place components of types i, j and k using heads 1, 2 and 3, respectively. Suppose, for instance, that these components are to be placed in the order j, i, k. Such an operation can be decomposed as follows. First, the arm moves until head 1 is positioned above a feeder of type i. Head 1 then picks a component i. Two more moves of the arm between the feeder slots allow heads 2 and 3 to pick components j and k. Next, the arm brings head 2 above the location where component j is to be placed, and the insertion is performed. The same operation is subsequently repeated for i and finally for k.

Some remarks are in order concerning such a pick-and-place round. Notice that the picking of the components must always be performed by head 1 first, then by head 2, then by head 3 (of course, we may decide in some rounds to use only one or two heads), whereas an arbitrary sequence may be selected for their placement. Once a head has been positioned by the arm above the required feeder slot or location, the time needed to pick or to place the corresponding component depends on the type of component, but is otherwise constant. Thus, on one machine, the only opportunities for a reduction of the total pick-and-place time reside in a clever sequencing of the operations and assignment of the feeders to feeder slots.

We have intentionally omitted many details in this brief description of the placement machines and of their functioning. For example, the insertion heads have to rotate to a predetermined angle before picking or placing components; some feeder slots or board locations are unreachable for certain heads; heads may be unavailable (e.g. broken) or may be assigned fixed equipment; the arm can only move in a limited number of directions; etc.

Some of these features (unreachable locations, unavailable heads, etc.) can be easily introduced into our models by setting variables to fixed values, thus resulting in a simplification of these models. Others will be implicitly incorporated in the models. For instance, parameters of the models such as the pick-and-place time or the travel time between board locations will be assumed to take into account the rotation of the heads and the restricted moves of the arm. Of course, there remains a possibility that these characteristics could be exploited explicitly to improve the performance of the machines, but we did not attempt to do so.

3. The throughput rate optimization problem

With this description of the technological constraints, we can now state a *global throughput rate optimization* problem as follows. Given the specifications of a PCB and of *M* placement machines, determine:

(1) an assignment of the components to the *M* machines;

(2) for each machine, an assignment of feeders to feeder slots;

(3) for each machine, a sequence of pick-and-place rounds, each round consisting itself of a sequence of at most three component locations among those assigned to the machine in step (1);

(4) for each machine and for each pick-and-place round, an assignment of equipment to heads.

These decisions are to be made so that the PCB can be mounted using all *M* machines, and so as to minimize the processing time on the *bottleneck machine* (i.e. the machine with the longest processing time).

In our solution of this problem, we shall also take into account a secondary criterion, dictated by cost considerations. Because feeders are rather expensive, it appears desirable (at least, in the practical situations that we encountered) to minimize the total number of feeders used. Ideally, thus, all components of a same type should be processed by one machine. In section 4.2, we shall show how this criterion can be accommodated.

This formulation of the throughput rate optimization problem is patterned after a (confidential) report of the Philips Center for Quantitative Methods (CQM [7]; see also van Laarhoven and Zijm [12]). This report proposes a hierarchical decomposition of the problem, and heuristics for the resulting subproblems. Our decomposition, as well as all heuristics presented in the next two sections, are different from CQMs. Our heuristics, in particular, rely more explicitly on the precise mathematical modelling of the subproblems.

The throughput rate optimization problem is also mentioned by Ball and Magazine [2], under somewhat simpler technological conditions. In particular, each machine has but one pick-and-place head. The authors investigate in detail only the sequencing of pick-and-place operations over one machine (i.e. our step (3) above).

Leipälä and Nevalainen [14] discuss our steps (2) and (3) for a different type of one-head machines.

Ahmadi et al. [1] consider the case of one machine featuring two heads. They address subproblems (2), (3) and (4), but their technological constraints are very different from ours, and their models do not seem to be directly applicable in our framework.

In the next two sections, we describe our approach to the throughput rate optimization problem. This approach is based on a decomposition of the global problem into the following list of subproblems (which thus refines the original formulation (1)–(4) given before):

(A) determine how many components each machine must mount, and with what equipment;

(B) assign feeder types to machines;

(C) determine which components each head must mount;

(D) cluster the locations into subsets of size at most three, to be processed in one pick-and-place round;

(E) determine the sequence of pick-and-place operations to be performed by each machine;

(F) assign the feeders to feeder slots.

Subproblems (A) and (B) in this list together answer question (1) and part of question (4) above. Our main concern in solving these two subproblems will be to achieve an approximate balance of the workload over the line of machines. This will be done in section 4.

Subproblems (C), (D), (E) and (F) address the scheduling of individual machines, and are dealt with in section 5.

In our computer experiments, the sequence of subproblems (A)–(F) is solved hierarchically in a single pass (except for (E) and (F); see section 5). It may be possible to use an iterative solution procedure, and to exploit the solution of certain subproblems in order to revise previous ones. We have not further explored these possibilities.

4. Workload balancing

4.1. SUBPROBLEM (A)

4.1.1. The model

We proceed in this phase to a preliminary distribution of the workload over the machine line, based on the number of equipment changes for each head and on a rough estimate of the time needed to mount each component. The latter estimate is computed as follows.

In section 1, we have seen that the time needed to mount a component of type t ($t = 1, \dots, T$) consists of two terms: a variable term measuring the travel time of the head, and a constant term p_t representing the total time spent to pick the component when the head is directly above feeder t, plus the time to place the component when the head is above the desired location.

Let v_t be an estimate of the first variable term; then, $v_t + p_t$ is an estimate of the mounting time required by each component of type t. Notice that, in practice, a reasonable value for v_t does not appear too difficult to come by, e.g. by evaluating the average time required for the arm to travel from feeder slots to mounting locations. The solution of the model given below does not appear to be very sensitive to the exact value of v_t. (In our computer experimentations, we used a constant value v for all v_t, $t = 1, \ldots, T$.) Otherwise, solving the model for a few alternative values of v_t $(t = 1, \ldots, T)$ provides different initial solutions for the subsequent phases of the procedure. If necessary, after all subproblems (A)–(F) have been solved, a solution to the global problem can be used to adjust the values v_t and reiterate the whole solution procedure.

Define now two component types to be *equivalent* if the quantity $v_t + p_t$ is the same for both types, and if both types can be handled by precisely the same equipment. This relation induces a partition of the set of components into C *classes*, with each class containing components of equivalent types.

We are now almost ready to describe our model. We first introduce a few more parameters:

Q = number of available equipments;

for $c = 1, \ldots, C$,

B_c = number of components in class c;

w_c = common value of $v_t + p_t$ for the types represented in class c;

$Q(c)$ = set of equipments which can handle the components in class c;

for $h = 1, \ldots, 3M$,

E_h = time required by an equipment change for head h.

The decision variables are: for $c = 1, \ldots, C$, for $m = 1, \ldots, M$, for $h = 1, \ldots, 3M$, for $q = 1, \ldots, Q$:

x_{cm} = number of components of class c to be mounted by machine m;

z_{mq} = 1 if machine m uses equipment q;
= 0 otherwise;

r_h = number of equipment changes required for head h;

W = estimated workload of the bottleneck machine.

The optimization model for subproblem (A) is:

($\mathrm{M_A}$) minimize W

subject to
$$\sum_{m=1}^{M} x_{cm} = B_c \qquad (c = 1, \ldots, C), \tag{1}$$

$$x_{cm} \le B_c \sum_{q \in Q(c)} z_{mq} \qquad (c = 1, \ldots, C;\ m = 1, \ldots, M), \tag{2}$$

$$\sum_{q=1}^{Q} z_{mq} \le \sum_{h=3m-2}^{3m} r_h + 3 \quad (m = 1, \ldots, M), \tag{3}$$

$$W \geq \sum_{c=1}^{C} w_c x_{cm} + \sum_{h=3m-2}^{3m} E_h r_h \qquad (m = 1, \ldots, M), \tag{4}$$

$$x_{cm} \geq 0 \text{ integer} \qquad (c = 1, \ldots, C;\ m = 1, \ldots, M), \tag{5}$$

$$z_{mq} \in \{0, 1\} \qquad (m = 1, \ldots, M;\ q = 1, \ldots, Q), \tag{6}$$

$$r_h \geq 0 \text{ integer} \qquad (h = 1, \ldots, 3M). \tag{7}$$

Constraints (1) express that all components must be mounted. Constraints (2) ensure that machine m is assigned at least one of the equipments in $Q(c)$ when x_{cm} is nonzero. Constraints (3) together with (4), (7) and the minimization objective, impose that the number of equipment changes on each machine be equal to the number of equipments used minus three, or equal to zero if the latter quantity becomes negative. The right-hand side of (4) evaluates the processing time on machine m (we assume here that the time needed to bring a new PCB onto the worktable, after completion of the previous one, is always larger than the time required for an equipment change). Thus, at the optimum of ($\mathrm{M_A}$), W is equal to the maximum of these processing times.

Two comments are in order concerning this model. First, we could have formulated a similar model using variables x_{km} instead of x_{cm}, with the index k running over all component locations, from 1 to N. The advantage of aggregating the components into classes is that the number of variables is greatly reduced, and that some flexibility remains for the exact assignment of operations to heads. This flexibility will be exploited in the solution of further subproblems. Second, observe that we do not impose any constraint on the number of feeder slots required by a solution of ($\mathrm{M_A}$). This could, in principle, be done easily, e.g. as in the partitioning model of Ahmadi et al. [1], but requires the introduction of a large number of new variables, resulting again from the disaggregation of classes into types. From a practical point of view, since we always allocate at most one feeder of each type per machine (remember the secondary criterion expressed in section 3), the number of feeder slots never appears to be a restrictive factor; hence, the solutions of ($\mathrm{M_A}$) are implementable.

In practice, the number of equipments needed to mount all components is often smaller than the number of heads available. When this is the case, we can in general safely assume that no change of equipment will be performed in the optimal solution of ($\mathrm{M_A}$) (since E_h is very large). We may then replace ($\mathrm{M_A}$) by a more restrictive model, obtained by fixing $r_h = 0$ for $h = 1, \ldots, 3M$.

4.1.2. *Complexity and solution of model* ($\mathrm{M_A}$)

Every instance of the well-known set-covering problem can be polynomially transformed to an instance of ($\mathrm{M_A}$) with $M = 1$, which implies that model ($\mathrm{M_A}$) is already NP-hard when only one machine is available (we assume the familiarity of the reader with the basic concepts of complexity theory; see, for example, Garey and

Johnson [10] or Nemhauser and Wolsey [15]; the proofs of all the complexity results can be found in Crama et al. [5]).

In spite of this negative result, obtaining solutions of good quality for $(\mathbf{M}_A)$ turns out to be easy in practical applications. To understand this better, notice that the number of variables in these applications is usually small. The real-world machine line which motivated our study features three machines. A typical PCB may require the insertion of a few hundred components, but these fall into five to ten classes. The number of equipments needed to mount the board (after deletion of a few clearly redundant ones) seems rarely to exceed five. So, we have to deal in $(\mathbf{M}_A)$ with about 10 to 30 zero–one variables and 15 to 50 integer variables.

In view of these favourable conditions, we take a two-phase approach to the solution of $(\mathbf{M}_A)$. In the first phase, we consider the relaxation of $(\mathbf{M}_A)$ obtained by omitting the integrality requirement on the x-variables (in constraints (5)). The resulting mixed-integer program is easily solved by any commercial branch-and-bound code (one may also envision the development of a special code for this relaxed model, but this never appeared necessary in this context).

In the second phase, we fix all r- and z-variables of $(\mathbf{M}_A)$ to the values obtained in the optimal solution of the first phase. In this way, we obtain a model of the form:

$$
\begin{aligned}
(\mathbf{M}'_A) \quad & \text{minimize } W \\
& \text{subject to} \quad \sum_{m=1}^{M} x_{cm} = B_c \qquad (c = 1, \ldots, C), \\
& \qquad W \geq \sum_{c=1}^{C} w_c x_{cm} + W_m \quad (m = 1, \ldots, M), \\
& \qquad x_{cm} \geq 0 \text{ integer} \qquad (c = 1, \ldots, C;\ m = 1, \ldots, M),
\end{aligned}
$$

where some variables x_{cm} are possibly fixed to zero (by constraints (2) of $(\mathbf{M}_A)$), and W_m is the total time required for equipment changes on machine m $(m = 1, \ldots, M)$.

In practice, model $(\mathbf{M}'_A)$ is again relatively easy to solve (even though one can show by an easy argument that $(\mathbf{M}'_A)$ is NP-hard). If we cannot solve it optimally, then we simply round up or down the values assumed by the x-variables in the optimal solution of the first phase, while preserving equality in the constraints (1).

In our implementation of this solution approach, we actually added a third phase to the procedure. The goal of this third phase is twofold: (1) to improve the heuristic solutions found in the first two phases; (2) to generate alternative "good" solutions of $(\mathbf{M}_A)$, which can be used as initial solutions for the subsequent subproblems of our hierarchical approach.

Two types of ideas are applied in the third phase. On the one hand, we modify "locally" the solutions delivered by phase 1 or 2, e.g. by exchanging the equipment of two machines, or by decreasing the workload of one machine at the expense of some other machine. On the other hand, we slightly modify model (M_A) by imposing an upper bound on the number of components assigned to each machine, and we solve this new model.

Running the third phase results in the generation of a few alternative solutions associated with reasonable low estimates of the bottleneck workload.

4.2. SUBPROBLEM (B)

4.2.1. The model

At the beginning of this phase, we know how many components of each class are to be mounted on each machine, i.e. the values of the variables x_{cm} in model (M_A). Our goal is now to disaggregate these figures and to determine how many components of each type must be handled by each machine. The criterion to make this decision will be the minimization of the number of feeders required (this is the secondary criterion discussed in section 3).

So, consider now an arbitrary (but fixed) class c. Reorder the types of components so that the types of components contained in class c are indexed by $t = 1, \dots, R$. Recall that N_t is the total number of components of type t to be placed on the board for all t. To simplify our notation, we also let $X_m = x_{cm}$ denote the number of components of class c to be mounted by machine m. So, $\sum_{t=1}^{R} N_t = \sum_{m=1}^{M} X_m = B_c$. We define the following decision variables: for $t = 1, \dots, R$, for $m = 1, \dots, M$;

u_{tm} = number of components of type t to be mounted by machine m;

v_{tm} = 1 if a feeder of type t is required on machine m;

= 0 otherwise.

Our model for subproblem (B) is:

$$(M_B) \quad \text{minimize} \sum_{t=1}^{R} \sum_{m=1}^{M} v_{tm}$$

$$\text{subject to} \sum_{t=1}^{R} u_{tm} = X_m \qquad (m = 1, \dots, M),$$

$$\sum_{m=1}^{M} u_{tm} = N_t \qquad (t = 1, \dots, R),$$

$$u_{tm} \le \min(X_m, N_t) v_{tm} \quad (t = 1, \dots, R;\ m = 1, \dots, M),$$

$$u_{tm} \ge 0 \text{ integer} \qquad (t = 1, \dots, R;\ m = 1, \dots, M),$$

$$v_{tm} \in \{0, 1\} \qquad (t = 1, \dots, R;\ m = 1, \dots, M).$$

Model ($\mathbf{M}_B$) is a so-called pure fixed-charge transportation problem (see Fisk and McKeown [8], Nemhauser and Wolsey [15]).

Another way of thinking about model ($\mathbf{M}_B$) is in terms of machine scheduling. Consider R jobs and M machines, where each job can be processed on any machine. Job t needs a processing time N_t ($t = 1, \ldots, R$) and machine m is available only in the interval $[0, X_m]$ ($m = 1, \ldots, M$). Recall that $\sum_{t=1}^{R} N_t = \sum_{m=1}^{M} X_m$. So, if preemption is allowed, there exists a feasible schedule requiring exactly the available time of each machine. Model ($\mathbf{M}_B$) asks for such a schedule minimizing the number of preempted jobs (in this interpretation, $v_{tm} = 1$ if and only if job t is processed on machine m).

4.2.2. Complexity and solution of model ($\mathbf{M}_B$)

The well-known partition problem can be polynomially transformed to model ($\mathbf{M}_B$), which implies that ($\mathbf{M}_B$) is NP-hard.

Model ($\mathbf{M}_B$) can be tackled by a specialized cutting-plane algorithm for fixed-charge transportation problems (Nemhauser and Wolsey [15]), but we choose to use instead a simple heuristic. This heuristic consists of repeatedly applying the following rule, until all component types are assigned:

Rule: Assign the type (say t) with largest number N_t of components to the machine (say m) with largest availability X_m; if $N_t \leq X_m$, delete type t from the list and reduce X_m to $X_m - N_t$; otherwise, reduce N_t to $N_t - X_m$ and X_m to 0.

Clearly, this heuristic always delivers a feasible solution of ($\mathbf{M}_B$), with value exceeding the optimum of ($\mathbf{M}_B$) by at most $M - 1$ (since, of all the component types assigned to a machine, at most one is also assigned to another machine). In other words, for a class c containing R component types, the heuristic finds an assignment of types to machines requiring at most $R + M - 1$ feeders. This performance is likely to be quite satisfactory, since R is usually large with respect to M.

In situations where duplication of feeders is strictly ruled out, i.e. where *all* components of one type *must* be mounted by the same machine, we replace the heuristic rule given above by:

Modified rule: Assign the type (say t) with largest number N_t of components to the machine with largest availability X_m; delete type t from the list; reduce X_m to $\max(0, X_m - N_t)$.

Of course, this modified rule does not, in general, produce a feasible solution of ($\mathbf{M}_B$). In particular, some machine m may have to mount more components of class c than the amount X_m determined by subproblem (A), and the estimated workload W of the bottleneck machine may increase. In such a case, we continue with the solution supplied by the modified rule. A possible increase in estimated workload is the price to be paid for imposing more stringent requirements on the solution.

Before proceeding to the next phase, i.e. the scheduling of individual machines, we still have to settle one last point concerning the distribution of the workload over the machines. Namely, the solution of model ($\mathbf{M}_\mathrm{B}$) tells us how many components of each type must be processed by each machine (namely, u_{tm}), but not which ones. Since the latter decision does not seem to affect very much the quality of our final solution, we neglect to give many details about its implementation here. Let us simply mention that we rely on a model aiming at an even dispersion of the components over the PCB for each machine. The dispersion is measured as follows: we subdivide the PCB into cells, and we sum up the discrepancies between the expected number of components in each cell and their actual number. It is then easy to set up an integer linear programming problem, where the assignment of components to machines is modelled by 0–1 variables, and the objective corresponds to dispersion minimization. The optimal solution of this problem determines completely the final workload distribution.

5. Scheduling of individual machines

In this section, we concentrate on one individual machine (for simplicity, we henceforth omit the machine index). Given by subproblem (B) are the locations (say $1, \ldots, N$) of the components to be mounted by this machine and their types $(1, \ldots, T)$. Given by subproblem (A) are the equipments $(1, \ldots, Q)$ to be used by the machine, and the number r_h of equipment changes per head.

5.1. SUBPROBLEM (C)

5.1.1. The model

Our current goal is to determine the distribution of the workload over the three heads of the machine (a similar "partitioning" problem is treated by Ahmadi et al. [1], under quite different technological conditions). This will be done so as to minimize the number of trips made by the heads between the feeder slots and the PCB. In other words, we want to minimize the maximum number of components mounted by a head. In general, this criterion will only determine how many components each head must pick and place, but not which ones. The latter indeterminacy will be lifted by the introduction of a secondary criterion, to be explained at the end of this subsection.

Here, we are going to use a model very similar to ($\mathbf{M}_\mathrm{A}$). Since we are only interested in the number of components mounted by each head, let us redefine two components as *equivalent* if they can be handled by the same equipment (compare with the definition used in subsection 4.1). This relation determines C classes of equivalent components. As for the subproblem ($\mathbf{M}_\mathrm{A}$), we let, for $c = 1, \ldots, C$:

B_c = number of components in class c;

$Q(c)$ = set of equipment which can handle the components in class c.

We use the following decision variables: for $c = 1, \ldots, C$, for $h = 1, 2, 3$, for $q = 1, \ldots, Q$:

x_{ch} = number of components of class c to be mounted by head h;
z_{hq} = 1 if head h used equipment q;
= 0 otherwise;
V = number of components mounted by the bottleneck head.

The model for subproblem (C) is:

$$(\mathrm{M_C}) \qquad \text{minimize } V$$

$$\text{subject to } \sum_{h=1}^{3} x_{ch} = B_c \qquad (c = 1, \ldots, C),$$

$$x_{ch} \le B_c \sum_{q \in Q(c)} z_{hq} \quad (c = 1, \ldots, C;\ h = 1, 2, 3),$$

$$\sum_{q=1}^{Q} z_{hq} = r_h + 1 \qquad (h = 1, 2, 3),$$

$$V \ge \sum_{c=1}^{C} x_{ch} \qquad (h = 1, 2, 3),$$

$$x_{ch} \ge 0 \text{ integer} \qquad (c = 1, \ldots, C;\ h = 1, 2, 3),$$

$$z_{hq} \in \{0, 1\} \qquad (h = 1, 2, 3;\ q = 1, \ldots, Q).$$

(Recall that $r_h + 1$ is the number of equipments allocated to head h by model $(\mathrm{M_A})$.)

5.1.2. Complexity and solution of model $(\mathrm{M_C})$

Again, the partition problem is easily transformed to model $(\mathrm{M_C})$, implying that the problem is NP-hard.

Moreover, as was the case for $(\mathrm{M_A})$, model $(\mathrm{M_C})$ is actually easy to solve in practice, due to the small number of variables. Here, we can use the same type of two-phase approach outlined for $(\mathrm{M_A})$.

As mentioned earlier, the solution of $(\mathrm{M_C})$ does not identify which components have to be mounted by each head. To answer the latter question, we considered different models taking into account the dispersion of the components over the board. However, it turned out empirically that a simple assignment procedure performed at least as well as the more sophisticated heuristics derived from these models. We now describe this procedure.

Consider a coordinate axis parallel to the arm along which the three heads are mounted. We orient this axis so that the coordinates of heads 1, 2 and 3 are of the form X, $X + k$ and $X + 2k$, respectively, where k is the distance between two heads ($k > 0$). Notice that X is variable, whereas k is fixed, since the arm cannot rotate.

The idea of our procedure is to assign the component locations with smallest coordinates to head 1, those with largest coordinates to head 3, and the remaining ones to head 2. Since this must be done within the restrictions imposed by ($\mathbf{M}_C$), let us consider the values x_{ch} obtained by solving ($\mathbf{M}_C$). Then, for each c, the components of class c to be mounted by head 1 are chosen to be the x_{c1} components with smallest coordinates among all components of class c. Similarly, head 3 is assigned the x_{c3} components with largest coordinates among the components of class c, and head 2 is assigned the remaining ones.

As mentioned before, this heuristic provided good empirical results. The reason for this good performance may be sought in the fact that the inter-head distance k is of the same order of magnitude as the length of a typical PCB. Thus, our simple-minded procedure tends to minimize the distance travelled by the heads.

5.2. SUBPROBLEM (D)

5.2.1. The model

For simplicity, we first consider the case where every head has been assigned exactly one piece of equipment (i.e. $r_1 = r_2 = r_3 = 0$ in model ($\mathbf{M}_C$)). Thus, at this point, the components have been partitioned into three *groups*, with group h containing the G_h components to be mounted by head h ($h = 1, 2, 3$). Let us further assume that $G_1 = G_2 = G_3 = G$ (if this is not the case, then we add a number of "dummy" components to the smaller groups). We know that G is also the minimum number of pick-and-place rounds necessary to mount all these components. We are now going to determine the composition of these rounds, with a view to minimizing the total travel time of the arm supporting the heads.

Suppose that the components in each group have been (arbitrarily) numbered $1, \ldots, G$. Consider two components i and j belonging to different groups, and assume that these components are to be mounted successively, in the same round. We denote by d_{ij} the time necessary to reposition the arm between the insertions of i and j. For instance, if i is in group 1, j is in group 2, and i must be placed before j, then d_{ij} is the time required to bring head 2 above the location of j, starting with head 1 above i.

For a pick-and-place round involving three components i, j, k, we can arbitrarily choose the order in which these components are mounted (see section 2). Therefore, an underestimate for the travel time of the arm between the first and the third placements of this round is given by:

(i) $d_{ijk} = \min\{d_{ij} + d_{jk}, d_{ik} + d_{kj}, d_{ji} + d_{ik}\}$ if none of i, j, k is a dummy;

(ii) $d_{ijk} = d_{ij}$ if k is a dummy;

(iii) $d_{ijk} = 0$ if at least two of i, j, k are dummies.

Let us introduce the decision variables u_{ijk}, for $i, j, k \in \{1, \ldots, G\}$, with the interpretation:

u_{ijk} = 1 if components i, j and k, from groups 1, 2 and 3, respectively, are mounted in the same round;
= 0 otherwise.

Then, our optimization model for subproblem (D) is:

$$(\mathrm{M_D}) \quad \text{minimize} \sum_{i=1}^{G} \sum_{j=1}^{G} \sum_{k=1}^{G} d_{ijk}\, u_{ijk}$$

$$\text{subject to} \sum_{i=1}^{G} \sum_{j=1}^{G} u_{ijk} = 1 \quad (k = 1, \ldots, G),$$

$$\sum_{i=1}^{G} \sum_{k=1}^{G} u_{ijk} = 1 \quad (j = 1, \ldots, G),$$

$$\sum_{j=1}^{G} \sum_{k=1}^{G} u_{ijk} = 1 \quad (i = 1, \ldots, G),$$

$$u_{ijk} \in \{0, 1\} \quad (i, j, k = 1, \ldots, G).$$

An optimal solution of $(\mathrm{M_D})$ determines G clusters, of three components each, such that the sum of the (underestimates of the) travel times "within clusters" is minimized.

In cases where some or all of the heads have been assigned more than one piece of equipment in model $(\mathrm{M_A})$, we adapt our approach as follows. Let q_h be the first piece of equipment to be used by head h and G_h be the number of components which can be handled by q_h among those to be mounted by head h ($h = 1, 2, 3$). Say, for instance, that $G_1 \leq G_2 \leq G_3$. We can now set up a model similar to $(\mathrm{M_D})$ for the clustering of these $G_1 + G_2 + G_3$ components. Any feasible solution of this model determines exactly G_1 clusters containing no dummy components. These clusters correspond to our first G_1, pick-and-place rounds, to be performed by equipment q_1, q_2 and q_3. Next, q_1 is replaced by a new piece of equipment q_4, and the process can be repeated with q_4, q_2 and q_3.

5.2.2. *Complexity and solution of* $(\mathrm{M_D})$

Model $(\mathrm{M_D})$, with arbitrary coefficients d_{ijk}, has been studied in the literature under the name of the three-dimensional assignment problem. The problem is known to be NP-hard (see Garey and Johnson [10]). However, observe that, in our case, the coefficients d_{ijk} are of the very special type defined by (i)–(iii). Moreover, the travel times d_{ij} $(i, j = 1, \ldots, G)$ are themselves far from arbitrary; in particular, they satisfy the triangle inequality: $d_{ij} \leq d_{ik} + d_{kj}$ for $i, j, k = 1, \ldots, G$. However, even under these added restrictions, model $(\mathbf{M}_\mathbf{D})$ remains NP-hard (Crama and Spieksma [6]).

A number of heuristic and exact algorithms have been proposed to solve the three-dimensional assignment problem (see, for example, Frieze and Yadegar [9] and the references therein). In view of the role of $(\mathbf{M}_\mathbf{D})$ as a subproblem in the hierarchy (A)–(F), and of the special structure of its cost coefficients, we opt here for a specialized heuristic procedure.

Our heuristic works in two phases. We start by solving an (ordinary) assignment problem, obtained by disregarding the components of the third group. Thus, we solve:

$$\textbf{(AP1)} \qquad \text{minimize} \sum_{i=1}^{G} \sum_{j=1}^{G} d_{ij}\, u_{ij}$$

$$\text{subject to} \sum_{j=1}^{G} u_{ij} = 1 \qquad (i = 1, \ldots, G),$$

$$\sum_{i=1}^{G} u_{ij} = 1 \qquad (j = 1, \ldots, G),$$

$$u_{ij} \in \{0, 1\} \qquad (i, j = 1, \ldots, G),$$

where $d_{ij} = 0$ if either i or j is a dummy. An optimal solution u^* of **(AP1)** can be computed in time $O(G^3)$ (Papadimitriou and Steiglitz [16]).

Now let $A = \{(i, j) : u^*_{ij} = 1\}$. Thus, A is the set of pairs (i, j) matched by the solution of **(AP1)**. The second phase of our heuristic consists of assigning the (previously disregarded) components of the third group to the pairs in A. Formally, we solve:

$$\textbf{(AP2)} \qquad \text{minimize} \sum_{(i,j) \in A} \sum_{k=1}^{G} d_{ijk}\, u_{ijk}$$

$$\text{subject to} \sum_{k=1}^{G} u_{ijk} = 1 \qquad ((i, j) \in A),$$

$$\sum_{(i,j) \in A} u_{ijk} = 1 \qquad (k = 1, \ldots, G),$$

$$u_{ijk} \in \{0, 1\} \qquad ((i, j) \in A;\ k = 1, \ldots, G).$$

The optimal solution of **(AP2)** can be obtained in time $O(G^3)$ and provides a heuristic solution of $(\mathbf{M}_\mathbf{D})$. Frieze and Yadegar [9] proposed a closely related heuristic for general three-dimensional assignment problems, and observed its good empirical performance.

Let β_3 denote the optimal value of (**AP2**). The notation β_3 is a reminder that, in the first phase of our heuristic, we arbitrarily decided to disregard the components from the third group. Of course, similar procedures could be defined, and corresponding bounds β_1 and β_2 would be derived, by initially disregarding the components from either group 1 or group 2.

In our computer implementations, we compute the three bounds β_1, β_2, β_3, and we retain the clustering of the components corresponding to the smallest bound. In Crama and Spieksma [6], it is proven that this bound is never worse than 4/3 times the optimal value for any instance of ($\mathbf{M}_\mathrm{D}$). The computer experiments reported there indicate that the practical performance of this heuristic is excellent.

5.3. SUBPROBLEM (E)

The solution of subproblem (D) has supplied a list $C_1, \ldots, C_G$ of clusters, with each cluster containing (at most) three components to be placed in the same round (if some heads must use more than one piece of equipment, then we successively consider several such lists, where each list consists of clusters which can be processed without equipment changes). Subproblem (E) asks for the sequence of pick-and-place operations to be performed by the machine, given this list of clusters.

This problem has been studied by Ball and Magazine [2], and Leipälä and Nevalainen [14], for machines featuring only one insertion head. In both papers, the authors observed that the decisions to be made in subproblem (E) are highly dependent on the assignment of feeders to feeder slots (i.e. on the solution of our subproblem (F)), and conversely. On the other hand, a model simultaneously taking into account both subproblems is far too complicated to be of any practical value.

We therefore choose an approach already suggested by Leipälä and Nevalainen [14]. Namely, we first solve subproblem (E); using this solution as input, we compute a solution of subproblem (F), which in turn is used to revise the solution of subproblem (E), and so on. This process is iterated until some stopping condition is verified.

5.3.1. The models

According to the previous discussion, we need two models for subproblem (E): the first one to be used when no feeder assignment is yet known, and the second one taking into account a given feeder assignment. In either case, we reduce (E) to the solution of a shortest Hamiltonian path problem (see Lawler et al. [13]) over the set of clusters $\{C_1, \ldots, C_G\}$: for $i, j = 1, \ldots, G$, we define a cost (travel time) $c(i, j)$ for processing C_i immediately before C_j; the problem is then to find a permutation $\sigma = (\sigma_1, \ldots, \sigma_G)$ of $\{1, \ldots, G\}$ which minimizes

$$c(\sigma) = \sum_{i=1}^{G-1} c(\sigma_i, \sigma_{i+1}). \tag{8}$$

The definition of $c(i, j)$ depends on the given feeder assignment (if any), as explained hereunder.

Consider first the situation where feeders are already assigned to feeder slots, and let C_i, C_j be two arbitrary clusters. In this case, the appropriate definition of $c(i, j)$ is given by CQM [7] as follows. Denote by l_1, l_2, l_3 the component locations in C_i, where l_h is to be processed by head h ($h = 1, 2, 3$). We assume that the feeder needed for l_h is in slot s_h ($h = 1, 2, 3$). Similarly, l_4 is the location to be processed by head 1 in cluster C_j, and slot s_4 contains the corresponding feeder (for simplicity, we assume that C_i and C_j consist of exactly three locations; obvious modifications of our description are required when this is not the case).

Suppose now for a moment that l_1, l_2 and l_3 are to be mounted in the order $\pi = (\pi_1, \pi_2, \pi_3)$, where (π_1, π_2, π_3) is a permutation of $\{1, 2, 3\}$. For this fixed order, we can easily compute the time (say, $c_{ij}(\pi)$) required to carry out the following operations: starting with head 1 above slot s_1, sequentially pick one component from each of s_1, s_2, s_3 using heads 1, 2, 3, respectively; mount $l_{\pi_1}, l_{\pi_2}, l_{\pi_3}$, in that order; bring head 1 above slot s_4.

Obviously, in an optimal pick-and-place sequence we would select the permutation π^* of $\{1, 2, 3\}$ which minimizes $c_{ij}(\pi)$. We accordingly define: $c(i, j) = c_{ij}(\pi^*)$.

Now, if σ is any permutation of $\{1, \ldots, G\}$, then $c(\sigma)$ (given by (8)) is the time required by a complete pick-and-place sequence processing the clusters in the order $(\sigma_1, \ldots, \sigma_G)$. The shortest Hamiltonian path problem with costs c_{ij} thus provides a natural model for subproblem (E). As a last remark on this model, notice that the computation of $c_{ij}(\pi)$ can be simplified by omitting from its definition those elements which are independent of π or σ. Namely, we can use a "modified $c_{ij}(\pi)$" defined as the time needed, starting with head 3 above s_3, to bring successively head π_1 above l_{π_1}, head π_2 above l_{π_2}, head π_3 above l_{π_3}, and finally head 1 above s_4.

Let us now return to the initial solution of (E), when the feeder positions are still unknown. Since this initial sequence will be modified by the subsequent iterations of our procedure, it does not seem necessary at this stage to look for a solution of very high quality (actually, one may even argue that an initial sequencing of lower quality is desirable since it provides more flexibility in the next phases of the procedure; see, for example, Leipälä and Nevalainen [14] for more comments along this line). Accordingly, we define the coefficients $c(i, j)$ for our initial travelling salesman problem as rough estimates of the actual travel times. We experimented with some possible definitions, which seem to lead to comparable results (in terms of the final solution obtained). One such definition is as follows. Let g_i and g_j be the centers of gravity of the clusters C_i and C_j, respectively. Let s be the feeder slot minimizing the total distance from g_i to s to g_j. Then, $c(i, j)$ is the time needed for the arm to travel this total distance.

5.3.2. Complexity and solution of the models

The shortest Hamiltonian path problem is closely related to the travelling saleman problem, and is well-known to be NP-hard, even when the costs $c(i,j)$ satisfy the triangle inequality (Lawler et al. [13]). Many heuristics have been devised for this problem, and we have chosen to experiment with two of the simplest: nearest neighbour (with all possible starting points) and farthest insertion which, respectively, run in $O(G^3)$ and $O(G^2)$ steps (we refer to Lawler et al. [13] for details on these procedures). Both heuristics produced results of comparable quality.

5.4. SUBPROBLEM (F)

5.4.1. The model

As input to this subproblem, we are given the types $(1,\dots,T)$ and the locations $(1,\dots,N)$ of the components to be mounted, where $(1,\dots,N)$ is the mounting sequence determined by the previous solution of subproblem (E). Our problem now is to allocate each feeder $1,\dots,T$ to one of the feeder slots $1,\dots,S$ so as to minimize the total mounting time (for the sake of clarity, we first assume that every feeder can be loaded in exactly one slot; we indicate later how our model can be modified when some feeders require two or more slots).

We use the decision variables v_{ts} $(t = 1,\dots,T;\ s = 1,\dots,S)$ with the interpretation:

$$v_{ts} = 1 \text{ if feeder } t \text{ is loaded in slot } s;$$
$$= 0 \text{ otherwise.}$$

These variables must obey the following restrictions, expressing that every feeder occupies exactly one slot, and no slot contains two feeders:

$$\sum_{s=1}^{S} v_{ts} = 1 \qquad (t = 1,\dots,T), \tag{9}$$

$$\sum_{t=1}^{T} v_{ts} \le 1 \qquad (s = 1,\dots,S), \tag{10}$$

$$v_{ts} \in \{0,1\} \quad (t = 1,\dots,T;\ s = 1,\dots,S). \tag{11}$$

Before describing the other elements of our model, we first introduce some terminological conventions. We say that a movement of the arm is a *feeder–board* movement if it occurs between the last picking and the first placing of the same round, or between the last placing of a round and the first picking of the next one. By contrast, a *feeder–feeder* movement takes place between two pickings of the same round.

Consider now a fixed solution v_{ts} ($t = 1, \ldots, T$; $s = 1, \ldots, S$) of (9)–(11). For the corresponding assignment of feeders to slots, the total mounting time of the PCB can be broken up into three terms:

(1) a term $\sum_{t=1}^{T} \sum_{s=1}^{S} a_{ts} v_{ts}$, where a_{ts} is the total time spent in feeder–board movements from or to feeder t, when feeder t is loaded in slot s; this term represents the total feeder–board travel time; notice that this value of each coefficient a_{ts} is completely determined by the technological features of the machine, and by the sequence of pick-and-place operations to be performed by the machine (i.e. by the solution of subproblem (E));

(2) a term $\sum_{p,t=1}^{T} \sum_{r,s=1}^{S} b_{prts} v_{pr} v_{ts}$, where b_{prts} is the total time spent in feeder–feeder movements between feeders p and t, when feeder p is in slot r and feeder t is in slot s; this term gives the total feeder–feeder travel time; here again, the coefficients b_{prts} are easily computed;

(3) a term accounting for all other operations (picking and placing of all components, and travel time between placements of the same round); for a fixed pick-and-place sequence, this term is independent of v_{ts}.

According to this discussion, our model for subproblem (F) can be formulated as:

$$(\mathrm{M_F}) \qquad \text{minimize} \sum_{t=1}^{T} \sum_{s=1}^{S} a_{ts} v_{ts} + \sum_{p,t=1}^{T} \sum_{r,s1}^{S} b_{prts} v_{pr} v_{ts}$$

subject to (9), (10), (11).

Problem ($\mathrm{M_F}$) is a quadratic assignment problem (see Burkard [3]). As mentioned earlier, this formulation can easily be modified to accommodate additional restrictions. For instance, if feeder t must occupy two slots, we reinterpret:

$$v_{ts} = 1 \text{ if feeder } t \text{ is loaded in slots } s \text{ and } s+1;$$
$$= 0 \text{ otherwise.}$$

Straightforward restrictions must then be added to (9)–(11) to preclude the assignment of any feeder to slot $s + 1$ when $v_{ts} = 1$. This can also be achieved while preserving the quadratic assignment structure of ($\mathrm{M_F}$) by raising all coefficients $b_{p,s+1,t,s}$ to very high values.

As a last remark on ($\mathrm{M_F}$), let us observe that this model boils down to a linear assignment problem for machines featuring only one insertion head. On the other hand, Leipälä and Nevalainen [14] proposed a quadratic assignment formulation of the feeder assignment subproblem (F) for another type of one-head machines. This discrepancy is obviously due to the different technologies.

5.4.2. *Complexity and solution of* ($\mathrm{M_F}$)

The quadratic assignment problem is well-known to be NP-hard and to be particularly difficult to solve exactly for values of T and S larger than twenty [3]. A typical instance of ($\mathrm{M_F}$) may involve as many as twenty feeder types and sixty slots, and hence must be tackled by heuristic methods.

For ($\mathrm{M_F}$), we have used a local improvement method, based on pairwise exchanges of feeders (see Burkard [3]). This procedure starts with an initial solution of (9)–(11), and applies either of the following steps as long as they improve the objective function value in ($\mathrm{M_F}$):

Step 1: move a feeder from its current slot to some empty slot;

Step 2: interchange the slot assignments of two feeders.

To determine an initial assignment of feeders to slots, we proceed in two phases. First, we solve the assignment problem ($\mathrm{M'_F}$) obtained by setting all coefficients b_{prts} to zero in ($\mathrm{M_F}$) (this amounts to disregarding the feeder–feeder movements of the arm). Let v^* be an optimal solution of ($\mathrm{M'_F}$).

Next, we consider those feeders (say $1, \ldots, P$) whose components are only picked by head 2. Observe that the associated variables v_{ts} ($t = 1, \ldots, P$; $s = 1, \ldots, S$) do not appear in the objective function of ($\mathrm{M'_F}$), since there are no feeder–board movements to or from these feeders (i.e. $a_{ts} = 0$ for $t = 1, \ldots, P$; $s = 1, \ldots, S$). Consequently, the value of these variables in v^* is conditioned only by the constraints (9)–(11), and may as well be random. In order to determine more meaningful values for these variables, we solve the restriction of ($\mathrm{M_F}$) obtained by setting $v_{ts} = v^*_{ts}$ for $t = P + 1, \ldots, T$ and $s = 1, \ldots, S$. It is easy to see that this again is a linear assignment problem, aiming at the minimization of the total feeder–feeder travel time under the partial assignment v^*_{ts} ($t = P + 1, \ldots, T$; $s = 1, \ldots, S$). The optimal solution of this problem, together with the values v^*_{ts} ($t = P + 1, \ldots, T$; $s = 1, \ldots, S$), provides the initial solution for the improvement procedure described above.

6. An example

In this section, we discuss the performance of our heuristics on a problem instance described in CQM [7]. The placement line under consideration consists of three machines. The third head is broken and unavailable on machine 3. The 258 components to be mounted on the PCB are grouped in 39 types (actually, the PCB is partitioned into three identical blocks of 86 components each; we shall make use of this peculiarity in the solution of subproblem (A)). Three distinct pieces of equipment suffice to handle all the component types; moreover, each type can be handled by exactly one of these three pieces of equipment.

For the sake of comparison, let us mention that CQM [7] evaluates to 74, 65 and 81 seconds, respectively, the mounting times required by the three machines for the actual operations sequence implemented by the plant (notice that this sequence is not known in full detail, and that these "plant times" appear to be underestimates). The hierarchical decomposition and the heuristics developed in CQM [7] produce a solution with mounting times of 68.41, 66.52 and 68.88 seconds for the three machines. A still better solution is obtained in CQM [7] after imposing the condition that the pieces of equipment used remain fixed, as in the plant situation. Under this condition, production times of 66.12, 65.25 and 65.47 are achieved on the three machines, i.e. an improvement of at least 18 percent of the bottleneck time with respect to the plant solution. To fully appreciate these figures, one should also know that a constant time of 106 seconds is needed for the pick-and-place operation alone, independently of the production sequence (see section 2). These unavoidable 106 seconds represent more than half of the total mounting time required by the CQM solutions.

6.1. SUBPROBLEM (A)

We now consider our subproblem (A). With a constant estimate of $v = 0.3$ (sec) for the travel time of the heads between two insertions, the components fall into five classes, characterized by the parameters in table 1.

Table 1
Parameters for subproblem (A)

Class	1	2	3	4	5
B_c	201	27	24	3	3
w_c	0.6	1.35	0.75	1.65	1.15
$Q(c)$	{1}	{2}	{3}	{3}	{3}

We set up model (M_A) with these parameters and $E_h = 2$ ($h = 1, \ldots, 8$) (and the obvious modifications implied by the unavailability of head 9). This model is easily solved by the approach described in subsection 4.1. Notice that the relaxation of (M_A) obtained by omitting the integrality requirement for the x-variables has several alternative optima. As expected, $r_h = 0$ ($h = 1, \ldots, 8$) in all these optimal solutions, i.e. equipment changes are ruled out.

As explained in subsection 4.1, the solutions found for subproblem (A) can be considered as alternative inputs for the subsequent subproblems in the decomposition. In the present case, most of these solutions led us to production plans with processing times of 66 to 68 seconds. To illustrate the next steps of our approach, we shall now concentrate on a specific solution of (M_A), derived as follows.

We mentioned before that our PCB consists of three identical blocks. So, rather than solving (M_A) for the complete board, we can solve first the model corresponding to one of the blocks, and eventually multiply all figures by 3. A workload distribution obtained in this way is displayed in table 2.

Table 2

Workload distribution

Machine	1	2	3
Equipment	1	1, 3	1, 2
x_{cm} = number of components of class c on machine m	$x_{11} = 102$	$x_{12} = 57$ $x_{32} = 24$ $x_{42} = 3$ $x_{52} = 3$	$x_{13} = 42$ $x_{23} = 27$

6.2. SUBPROBLEM (B)

Since all components of class 2 are to be handled by machine 3, and all components of classes 3, 4, 5 by machine 2, we see that the distribution shown in table 2 need only be further refined for class 1. Specifically, 28 component types are represented in class 1. The number of components of each type (1, . . . , 28) is given in table 3.

Table 3

Number of components of each type for subproblem (B)

Type	1	2	3	4	5	6	7	8	9	10	11	12	13	14
N_t	24	18	18	15	12	9	9	9	9	6	6	6	6	6
Type	15	16	17	18	19	20	21	22	23	24	25	26	27	28
N_t	6	6	3	3	3	3	3	3	3	3	3	3	3	3

The heuristic rule described in subsection 4.2 produces the assignment shown in table 4. Observe that each type is assigned to exactly one machine, and hence exactly one feeder of each type will be needed in the final solution (in particular, here the heuristic delivers an optimal solution of (M_B)).

Table 4

Assignment of component types to machines

Machine	Types
1	1, 2, 3, 5, 10, 11, 14, 17, 20, 23, 26
2	4, 6, 8, 12, 15, 18, 21, 24, 27
3	7, 9, 13, 16, 19, 22, 25, 28

6.3. SUBPROBLEM (C)

Since model (M_C) attempts to minimize the maximum workload of the heads (per machine), in this case we obviously find an assignment of the type given in table 5. The components to be mounted by heads 1, 2, 3, 4, 5 are further identified

Table 5

Assignment for subproblem (C)

Head	1	2	3	4	5	6	7	8
Equipment	1	1	1	1	1	3	1	2
No. of components	34	34	34	29	28	30	42	27

as explained at the end of subsection 5.2. In the present case, this amounts to assigning to head 1 all components of block 1, to head 2 all components of block 2, and to head 3 all components of block 3, among those previously assigned to machine 1.

6.4. SUBPROBLEM (D)

We now solve the three-dimensional assignment model (M_D) for each of the three machines. Since machine 3 has only two heads, (M_D) actually reduces to the assignment problem (**AP1**) for this machine, and hence can be solved exactly (optimal value: 3.26 sec).

For machines 1 and 2, we solve (M_D) using the heuristics described in subsection 5.2. For machine 1, these heuristics supply a very good clustering of the components (value: 4.95 sec), where each cluster simply contains corresponding components from each block of the PCB. For machine 2, we obtain a clustering with value 8.95 sec.

6.5. SUBPROBLEMS (E) AND (F)

These two subproblems are solved alternately and iteratively for each machine. On machine 2, for instance, the first Hamiltonian path (corresponding to travel times between centers of gravity of the clusters) has value 13.16 sec. An initial feeder assignment is obtained as in subsection 5.4. The pick-and-place sequence determined by this assignment and the first Hamiltonian path corresponds to a total feeder–board time of 14.10 sec and a total feeder–feeder time of 11.63 sec, for a total travel time of 25.73 sec.

The local improvement procedure is next applied to this initial solution. In *each iteration* of this procedure, we sequentially consider *all feeders*, and we attempt to perform one of the exchange steps 1 and 2 on each of them. After four iterations of the procedure, no more improving steps are found. The corresponding feeder–board and feeder–feeder times are, respectively, 14.68 sec and 8.62 sec, and hence the previous total travel time is improved to 23.30 sec.

Taking this feeder assignment into account, a revised Hamiltonian path with value 14.07 sec is computed. The feeder assignment is in turn modified, resulting in (after three iterations of the local improvement procedure) a total travel time of 22.94 sec. No better Hamiltonian path or assignment is found in the next solutions of subproblems (E) and (F). Therefore, we adopt this solution for machine 2.

Similar computations are carried out for the other machines. The pick-and-place sequences obtained in this way correspond to processing times of 63.83, 66.27 and 65.82 sec on machines 1, 2 and 3, respectively. These times are comparable to the best ones obtained by CQM.

Acknowledgements

We are grateful to P. van Laarhoven and H. Zijm for introducing the problem to us, and for providing us with the data [7].

References

[1] J. Ahmadi, S. Grotzinger and D. Johnson, Component allocation and partitioning for a dual delivery placement machine, Oper. Res. 36(1988)176–191.

[2] M.O. Ball and M.J. Magazine, Sequencing of insertions in printed circuit board assembly, Oper. Res. 36(1988)192–201.

[3] R.E. Burkard, Quadratic assignment problems, Eur. J. Oper. Res. 15(1984)283–289.

[4] J.A. Buzacott and D.D. Yao, Flexible manufacturing systems: A review of analytical models, Manag. Sci. 32(1986)890–905.

[5] Y. Crama, A.W.J. Kolen, A.G. Oerlemans and F.C.R. Spieksma, Throughput rate optimization in the automated assembly of printed circuit boards, Research Memorandum RM 89.034, Faculty of Economics and Business Administration, University of Limburg, Maastricht, The Netherlands (1989).

[6] Y. Crama and F.C.R. Spieksma, Approximation algorithms for three-dimensional assignment problems with triangle inequalities, Eur. J. Oper. Res, to appear.

[7] CQM, Philips Center for Quantitative Methods, Eindhoven, The Netherlands (1988).

[8] J. Fisk and P.G. McKeown, The pure fixed charge transportation problem, Naval Res. Logist. Quart. 26 (1979)631–641.

[9] A.M. Frieze and J. Yadegar, An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice, J. Oper. Res. Soc. 32(1981)989–995.

[10] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, New York, 1979).

[11] B. Korte, Applications of combinatorial optimization, in: *Mathematical Programming, Recent Developments and Applications*, ed. M. Iri and K. Tanabe (KTK Scientific Publ., Tokyo, 1989), pp. 1–55.

[12] P.J.M. van Laarhoven and W.H.M. Zijm, Production preparation and numerical control in PCB assembly, Memorandum M 90-09, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands (1990).

[13] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys (eds.), *The Traveling Salesman Problem* (Wiley, New York, 1985).

[14] T. Leipälä and O. Nevalainen, Optimization of the movements of a component placement machine, Eur. J. Oper. Res. 38(1989)167–177.

[15] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization* (Wiley, New York, 1988).

[16] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity* (Prentice Hall, Englewood Cliffs, NJ, 1982).