# New Models and Heuristics for Component Placement in Printed Circuit Board Assembly

Edmund K. Burke Peter I. Cowling Ralf Keuthen
University of Nottingham
School of Computer Science & IT
University Park, Nottingham NG7 2RD, UK
ekb@cs.nott.ac.uk pic@cs.nott.ac.uk rxk@cs.nott.ac.uk

## Abstract

*This paper considers an optimization problem arising in the automated manufacture of printed circuit boards. An essential part of this problem is to determine good component pick and place sequences as this can reduce the overall assembly time significantly. This component placement sequence problem can be modelled as a Travelling Salesman Problem, however complex machine and process specifications also arise in practical assembly problems. We describe the variety of placement machinery, providing new, detailed models for a variety of complex assembly problems. Of special interest is the case of a multi-headed placement machine for which we develop a new model and heuristic solution approaches.*

## 1. Introduction

The development of Surface Mount Technology **(SMT)** in Printed Circuit Board **(PCB)** assembly made it possible to mount a large number of components on a single circuit board. This rather new technology has been increasingly employed in the electronic manufacturing industry over the last decade and has displaced through-hole technology from its former leading position. The increase of components to be placed on a single board has made the reduction of assembly time probably the most important issue to further cut down production costs and increase productivity. The assembly of PCBs on a production line can be divided in three process planning problems:

1. the assignment of component types to machines,
2. the assignment of component types to component feeder locations for each machine,
3. the component placement sequencing for each machine.

Many publications have been devoted to this complex optimization problem and various models and techniques have been presented by Shellwell, Buzzcott and Magazine [18], Ball and Magazine [2], Brad et al. [3], Leipälä and Nevalainen [13], Kho and Ng [9], Moyer and Gupta [15], Foulds and Hamacher [5] and Leon and Peters [14] to name but a few.

In this paper we are going to consider a single component placement machine, assuming that the first problem, the allocation of component types to machines, has already been solved. In contrast to most former publications considering the problems (2) and (3), we have not separated these problems, we address the assignment of components to feeders as a complication of the component placement sequencing problem. As suggested in most papers, the component placement sequencing problem can be modelled as a *Travelling Salesman Problem* **(TSP)** [11] which is well known to be $NP-$hard. Finding good solutions is further complicated as placement machine specifications usually introduce additional restrictions which may be modelled as *additional side constraints* on the TSP. In this paper we will present some of the complicating factors arising for various placement machine types and suitable ways of modelling them. Of special interest to us has been the optimization of placement sequences for multi-headed placement machines. We have developed a new model and some heuristics for this complex sequencing problem.

The paper is structured as follows. In section 2 a range of component placement machines is described. Section 3 introduces suitable models for the component placement sequencing problem while considering complicating factors arising for machines described in section 2, introducing a new model for multi-headed placement machines. In section 4 we propose new heuristics to determine optimal sequences for multi-headed component placement machines. Section 5 presents our conclusions and gives a brief overview of planned future research.

## 2. Placement machinery in printed circuit board assembly

In this section we will describe some of the different types of placement machines used in printed circuit board assembly. These different types of machines vary significantly in their operation and so give rise to different complicating factors.

Generally, a component placement machine has either a moving board or a moving placement head. Further important parts of the placement machine are the feeder magazine where the components are stored for pick up and an automated tool changer which changes the tool in the placement head, since not all components can be picked-up using the same tool. A typical feeder magazine consists, according to Ahmadi et al. [1], of either several tape reel or vibratory ski slope feeders or both to ensure components are available for pick up. The placement head uses a variety of grippers or nozzles, for vacuum placement, and can rotate to orient the components for pick up and placement.

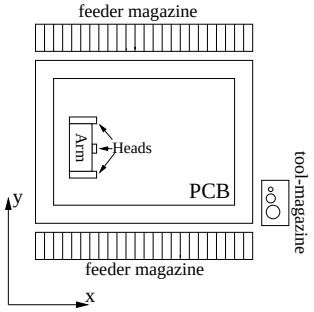


**Figure 1. Typical component placement machine**

For placement machines similar to the one shown in figure 1 we can distinguish between two types. The first type, as considered by Hong et al. [7], consists of a fixed board where the head moves in $x - y$ directions between feeder and insertion locations in order to perform the placement. The feeder magazines for machines with a moving head are usually fixed and the tool changer may be located around the board as in the figure.

The second type consists of a board moving in the $x - y$ plane where the head is generally not fixed but travels to a fixed pick up location and back to the fixed insertion position while the board moves to the insertion position. Since the head only moves to a fixed location to pick up components, magazine movements become necessary to supply the head with the required components. The tool changer is usually part of the feeder magazine and moves into place when a tool change is required. Either of these machine types may have multiple placement heads, each with different grippers or nozzles, mounted on an arm as shown in figure 1 and the number of feeder magazines and tool changers may vary but is usually either one or two. Some of the feeder magazines may be accessible only by some of the heads. A placement machine of the type described above is the *dual-delivery placement machine*, as discussed by Ahmadi et al. [1], which consists of a moving board, two heads and two moving feeder magazines with tool changers, where each head is only able to pick up components from its corresponding magazine on either side of the board.

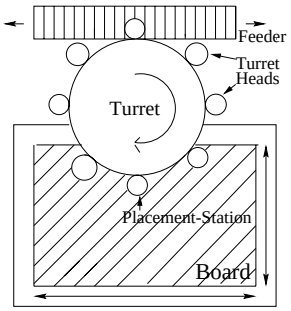


**Figure 2. High Speed Chip Shooter**

The *High Speed Chip Shooter* [16],[15] is fundamentally different to the machine types described above, see fig. 2. This placement machine uses a placement mechanism mounted on a rotating turret, with multiple placement heads, that rotates from a fixed pick up location to the insertion location. For this machine the single feeder magazine and the board must both move to ensure placement. According to Ng [16] the chip shooter has become a very popular machine in recent years, as it has proven to be faster than the types illustrated in fig. 1.

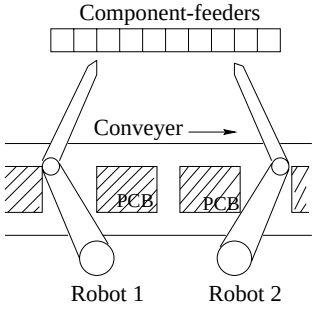


**Figure 3. 2-robot-arm placement machine**

A placement machine type with a completely different mechanical architecture has been considered by Park and Asada [17]. This placement machine consists of two robot

arms working independently to assemble boards moving on a moving conveyer line, as shown in fig. 3. The movements of the robot arms and the conveyer must be synchronized to make component placement possible. At any given time a number of boards are accessible for assembly, the feeder magazine is fixed and shared by both robots and each robot uses a single placement tool with no tool changes.

## 3. Modelling complicating factors arising in PCB assembly

This section introduces suitable models for general and machine specific complicating factors arising in PCB assembly. As mentioned before the component placement sequencing problem can be modelled as a Travelling Salesman Problem. To achieve this we need to define cities and distances. The common way to do this is to view the placement locations as the cities of the TSP and define distances by the time or the Euclidean distance traveled between two successive insertions, allowing the component pick up.

More formally the component placement sequencing problem can be modelled as follows. Let $\{c_1, c_2, \ldots, c_n\}$ be the finite set of placement locations, $d_{ij}$ denote the distance in terms of time or distance traveled between placements $c_i$ and $c_j$ and $\sigma : \{1, 2, \ldots n\} \rightarrow \{1, 2, \ldots n\}$ be a one to one mapping which orders the set of placements. Then the component placement sequencing problem is equivalent to finding $\sigma$ so as to

$$\underset{\sigma}{\text{minimize}} \quad \sum_{i=1}^{n-1} d_{\sigma(i),\sigma(i+1)} + d_{\sigma(n),\sigma(1)}. \tag{1}$$

The assignment of component types to feeder slots is, independently of the component placement machine in use, of substantial importance as it directly influences the time taken between two placements and thus the optimal sequence. The optimal permutation of components to the feeders has been addressed in various papers and different models have been proposed [1],[7]. Here, we will briefly introduce more general models for feeder assignment and sequencing of the placement machines described in the previous section.

Consider a single headed component placement machine, placement locations, each with its own component, $\{c_1, c_2, \ldots, c_n\}$, and $\{s_1, s_2, \ldots, s_f\}$ available feeder slots, let $\varrho : \{c_1, c_2, \ldots, c_n\} \rightarrow \{s_1, s_2, \ldots, s_f\}$ be a mapping assigning each placement location to a feeder slot and $d^{\varrho}_{i,j}$ be the time taken between finishing insertion of component $c_i$ and finishing insertion of component $c_j$ for the component assignment $\varrho$. We now have a two level optimization problem, to find $\sigma$ and $\varrho$ that:

$$\underset{\sigma,\ \varrho}{\text{minimize}} \sum_{i=1}^{n-1} d^{\varrho}_{\sigma(i),\sigma(i+1)} + d^{\varrho}_{\sigma(n),\sigma(1)} \ . \tag{2}$$

The aim of this optimization problem is to find an optimal movement sequence $\sigma$ for the placement head where in contrast to our initial problem (1) the distances between placement locations now depend on assignment of the components to the feeder slots. It is usually assumed that not more than one component type is assigned to a single feeder slot as critical components may need to be tested against a camera, as considered by Lee et al. [12] and Kumar and Li [10]. If they fail this test they are rejected and the next component of this type is picked from the feeder slot.

Later we consider the more general case where multiple pick ups for multi-headed placement machines, as discussed by Hong et al. [7] and Lee et al. [12], are taken into account. An important issue that has to be considered in modelling a printed circuit board assembly sequence are possible tool changes. Since the components to be inserted are usually not all of the same physical dimensions, changes of the gripper/nozzle are unavoidable. Tool changes are relatively time consuming compared to pick and place operations and should therefore only be performed when necessary. Therefore in most models tool changes are only carried out after all operations suitable to the current tool have actually been performed, so that often the problem can be decomposed into a separate TSP for each tool. Various approaches to model this problem for different machine types can be found in the literature [1],[7].

However, it is often the case that more than one of the different grippers/nozzles is able to pick up a certain component. This leads to a tool assignment and sequencing problem which can not be decomposed into separate TSPs for each tool. Let us assume that we have a set $\{t_1, t_2, \ldots, t_m\}$ of tools and let $\tau : \{c_1, c_2, \ldots, c_n\} \rightarrow \{t_1, t_2, \ldots, t_m\}$ be a function which assigns each placement location $c_i$ to a suitable tool $t_j$. In order to introduce a distance depending on component assignment function $\varrho$, as defined above, but also on the tool assignment function $\tau$, let $d^{\tau,\varrho}_{i,j}$ be the time it takes for the placement head to start with tool $\tau(c_i)$ at placement location $c_i$, perform a tool change if $\tau(c_i) \neq \tau(c_j)$, pick up the component at feeder slot $\varrho(c_j)$ and place it at location $c_j$. As tool changes are very time consuming, we will have $d^{\tau,\varrho}_{i,j} \ll d^{\tau,\varrho}_{k,l}$, if $\tau(c_i) = \tau(c_j)$ and $\tau(c_k) \neq \tau(c_l)$. This problem can now be modelled as a three level optimization problem where we aim to find $\sigma, \varrho$ and $\tau$ that

$$\underset{\sigma,\ \varrho\ \tau}{\text{minimize}} \sum_{i=1}^{n-1} d^{\varrho,\tau}_{\sigma(i),\sigma(i+1)} + d^{\varrho,\tau}_{\sigma(n),\sigma(1)}, \tag{3}$$

where $\sigma$ and $\varrho$ are defined as above. For this three level optimization problem the time between successive placements and therefore the sequence do not only depend on the assignment of components to feeder slots, as in (2), but also on the assignment of suitable tools to placement

locations. The mappings of tool assignment $\tau$ and sequence $\sigma$ will usually be closely related. Since the time taken to change a tool is so high relative to the component insertion and head travel time, $\tau$ will usually partition our tour $c_{\sigma(1)}, c_{\sigma(2)}, \ldots, c_{\sigma(n)}, c_{\sigma(1)}$ into disjoint segments, with each segment corresponding to a different tool.

The sequencing problem becomes more complicated when dealing with multi-headed machines. In this case pick ups may be unnecessary between some placements.

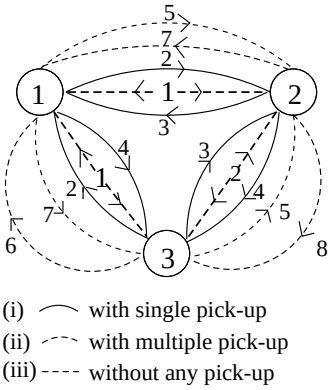


**Figure 4. Multiple distances for a 3-city problem**

Consider the three component problem in fig. 4 where our placement machine is assumed to have 2 heads. When considering two placement locations, we may have three distances, one corresponding to the time taken when we visit the feeder between placements to pick up a single component, a second distance where we pick up both of the other components before placement and a third in form of a short cut between placements belonging to component types following a multiple pick up, where we do not need to revisit the feeder between insertion. This problem is illustrated by a graph in fig. 4 for three placement locations. The various arcs and distances correspond for placement locations $c_i$ and $c_j$ to one of the following three pick and place scenarios (i) the head starts empty at location $i$, picks up the component to place at $i+1$ and moves to $i+1$ to place it, (ii) the head starts empty at $i$, picks up the next two components and places component $i+1$, and (iii) a pick up is not needed, since a multiple pick up has already been performed, and the head moves directly from $i$ to $i+1$ to place component $i+1$.

In order to optimize the tour while taking multiple pick ups into account, let us consider a placement machine with $h$ placement heads. This machine is able to pick up at most $h$ components between insertions. This may yield significant savings since we may arrange components on feeder magazines so that multiple components may be picked up simultaneously. Define the mapping of components to feeder slots $\varrho$ and placement locations to tools $\tau$ as above, and introduce a placement location $c_0$ which corresponds to the start/finish position of the placement head leading to a set of placement locations $C = \{c_0, c_1, \ldots, c_n\}$. We consider a generalization of a TSP tour, a *hypertour*, in order to model the case of a placement machine with multiple heads, as illustrated in fig. 1. Let $\mathcal{A} = (A_1, A_2, \ldots, A_k)$, where $A_i = (c_1^i, c_2^i, \ldots, c_{j_i}^i)$ is an ordered sequence with $c_j^i \in C$ and $c_j^i \neq c_k^i$ for $j \neq k$, be a hypertour, i.e. an overlapping set of sequences as illustrated in fig. 5, satisfying the following properties,

$$
\begin{aligned}
& 2 \leq |A_i| \leq h+1,\ i = 1, \ldots, k, \\
& |A_i \sqcap A_{i+1}| = 1,\ i = 1, \ldots, k-1, \\
& |A_k \sqcap A_1| = 1, \\
& c_{j_i}^i = c_1^{i+1},\ i = 1, \ldots, k-1, \\
& c_{j_k}^k = c_1^1 = c_0, \\
& A_i \sqcap A_j = \emptyset,\ i, j = 1, \ldots, k,\ |i-j| > 1 \\
& \quad \text{with } \{i, j\} \neq \{1, k\}, \\
& \bigsqcup_{j=1}^{k} A_j = \{c_0, c_1, c_2, \ldots, c_n\},
\end{aligned}
\tag{4}
$$

where $A_i \sqcup A_j$ denotes the set of all the elements in either $A_i$ or $A_j$ and $A_i \sqcap A_j$ denotes the set of elements common to both $A_i$ and $A_j$. In this model $A_i$ corresponds to a sequence starting with an empty head at position $c_1^i$, picking up all components corresponding to the locations $c_2^i, c_3^i, \ldots, c_{j_i}^i$ and then placing them in the given order finishing with an empty head at placement location $c_{j_i}^i$. Let $D^{\varrho,\tau} : (C^2 \cup C^3 \cup \ldots \cup C^{h+1}) \rightarrow \mathbb{R}$ be a function, where $D^{\varrho,\tau}(A_i)$ is the time taken to complete sequence $A_i$ for component assignment $\varrho$ and tool assignment $\tau$. To illustrate $D^{\varrho,\tau}$ with fixed $\varrho$ and $\tau$, the distances $D^{\varrho,\tau}$ corresponding to the graph in fig. 4, for all possible sequences $A_i$, are given in table 1.

**Table 1. Distances $D^{\varrho,\tau}$ for graph in fig. 4**

| $\lvert A_i \rvert = 2$ | | $\lvert A_i \rvert = 3$ | |
|---|---|---|---|
| set $A_i$ | $D^{\varrho,\tau}(A_i)$ | set $A_i$ | $D^{\varrho,\tau}(A_i)$ |
| $(c_1, c_2)$ | 2 | $(c_1, c_2, c_3)$ | 5+2=7 |
| $(c_1, c_3)$ | 4 | $(c_1, c_3, c_2)$ | 7+2=9 |
| $(c_2, c_1)$ | 3 | $(c_2, c_1, c_3)$ | 7+1=8 |
| $(c_2, c_3)$ | 4 | $(c_2, c_3, c_1)$ | 8+1=9 |
| $(c_3, c_1)$ | 2 | $(c_3, c_1, c_2)$ | 6+1=7 |
| $(c_3, c_2)$ | 3 | $(c_3, c_2, c_1)$ | 5+1=6 |

We can introduce a *Minimum Weight Hypertour Problem*, see fig. 5, which is a generalization of the TSP where we replace each edge of the TSP tour by a directed hyperedge. This leads us to a three level optimization problem where we aim to find a hypertour $\mathcal{A}$, an assignment $\varrho$ of

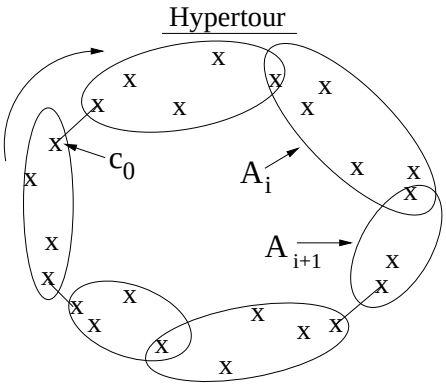


**Figure 5. Minimum Weight Hypertour Problem**

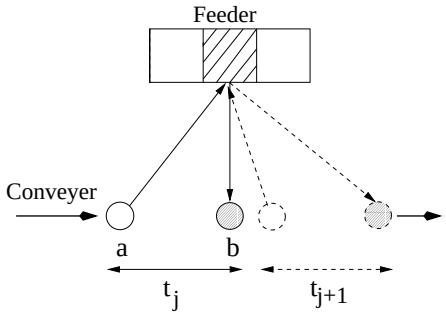


**Figure 6. Time dependency**

components to feeders and an assignment $\tau$ of placement locations to tools such that we

$$\underset{\mathcal{A},\ \varrho,\ \tau}{\text{minimize}} \sum_{i=1}^{k} D^{\varrho,\tau}(A_i)\,, \tag{5}$$

subject to the constraints (4) above. When dealing with a single headed placement machine each $A_i$ includes exactly two placement locations and we are back to the model in (3). We will consider a heuristic for this problem in the next section.

Another interesting restriction, provided by the arm movements of the dual-robot placement machine in fig. 3, is that interference of the arms must be strictly avoided. This introduces a collision avoidance constraint to our initial model. Even for two robot arms, collision avoidance turns out to be rather complicated. In order to keep distance between the arms we have to know the position of each arm at any time. Assuming the position of robot arm $i$ at time $t$ is given by $p_i(t)$ the collision avoidance constraint can be written as:

$$\min_t(\|p_1(t) - p_2(t)\|) \geq D_{min},$$

where $\|.\|$ is a suitable norm, for example the Euclidean distance, and $D_{min}$ denotes the minimal distance allowed between the robot arms. The expression for $p_i(t)$ may be derived from the tour. Note, that each robot arm is described here by a single position in space, which is a useful simplification of the actual problem. A second interesting problem for a machine of this type may be caused by the conveyer movement during assembly. The problem here is that the boards and hence the placement locations are moving on the conveyer while the component feeder is fixed. Then the distances between two placement locations depend upon time. The time dependency of the distance $d_{ij}$ leads to a *time dependent* Travelling Salesman Problem where the cost of travelling directly from city $i$ to $j$ at time $t$ is given by $d_{ij}(t)$. The graph in fig. 6 illustrates the distance change for a move from placement location $a$ to $b$ with corresponding pick up for times $t_j$ and $t_{j+1}$. The moving conveyer also introduces *time windows*, during which all placements on a board must be completed as each insertion on a board has to be performed during the fixed time period while the robot arms are able to reach it.

Some other constraints in printed circuit board assembly such as precedence constraints have been well studied in the literature [4].

## 4. Heuristics

In this section we are going to propose some heuristics to tackle the minimum weight hypertour problem, as described in the previous section, arising for multi-headed placement machines. Many efficient heuristics have been proposed for the Travelling Salesman Problem in the literature [8], [11]. However, these heuristics are not suitable for the model we are confronted with. The heuristics we have developed take their inspiration from two well known heuristics for the Travelling Salesman Problem which have been modified to be suitable for the hypertour problem. These heuristics are inspired by the *nearest neighbor* tour construction heuristic and the local search algorithm *k-opt*.

The hypertour problem as described previously consists of several important parts. The most common way to deal with a complicated model as this is to divide the original problem into several subproblems and solve them separately [7], [5]. However, this subdivision can give rise to a locally optimal solution which is far from the global optimum. Since these problems are all closely related and influence the optimality of the sequence, we went for a different approach by trying to solve these subproblems simultaneously.

In the following we are considering a $h-$headed component placement machine. We also assume for ease of expo-

sition that the number of different component types is equal to the number of feeder slots, i.e. that each component type is assigned to exactly one feeder slot, and that all placement heads have a "universal tool" capable of processing all component types.

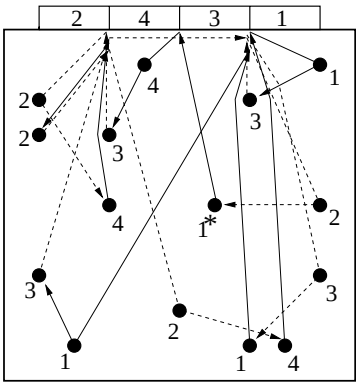


**Figure 7. Nearest Neighbor tour for a 2-headed placement machine**

The basic idea of the nearest neighbor tour construction heuristic for hypertours is to successively add hyperedges to the tour, where the elements of a hyperedge are selected in a way analogous to the nearest neighbor approach for the TSP. There are several different ways of defining the "nearest neighbor" for the hypertour problem. We detail one which has the benefit of low time complexity and outline its possible extension. We start with the placement head empty at some location $p_0$. Then we find a set $A$ of $h$ "near" placement locations, by moving from $p_0$ to the nearest feeder slot $s_0$ which still has components to be placed. The closest placement location $p_1$ which may still be fed from $s_0$ is added to $A$. We then consider picking up a second component from the next nearest feeder slot $s_1$ which still has components to be placed. The closest placement location to $p_1$ for a component from $s_1$ is added to $A$. We proceed thus until we have placement locations $p_0, p_1, p_2, \ldots, p_h$. Then we select optimal pick and place hyperedges $A_1, A_2, \ldots, A_h$ for these placements, where $A_i$ is the shortest hyperedge picked from the $\binom{h}{i} \cdot i!$ possible hyperedges of $i$ placement locations from $A$. In order to compare the length of these sequences and choose the optimal one as hyperedge we introduce the idea of a *weighted hyperedge length*. The length of a hyperedge $A$ is *weighted* by dividing it by the number of placements in $A$. The nearest neighbor tour construction algorithm can then be implemented as shown in table 2.

An example of this tour construction heuristic applied to a problem consisting of 15 placement locations of 4 different component types for a 2-headed component placement machine is shown in figure 7. In this figure the starting position is marked by a star and alternate hyperedges are distinguished by being connected by either solid or dashed arrows. Considering a total of $n$ components to place and $f$ component feeder slots, where usually $f \ll n$, the time complexity of this approach can be determined as follows. Starting at a placement location, the selection of the nearest feeder slot is of order $O(f)$ and then to choose the nearest placement location, after a feeder slot has been selected, is of order $O(n)$. This leads to a time complexity of $O(f+n)$ to choose the next component to be placed. To place all components this process has to be repeated $n$ times. This heuristic then has a time complexity of $O((n+f) \cdot n)$ or $O(n^2)$, the same as the nearest neighbor approach for the TSP.

A possible extension of this approach works as follows.

**Table 2. A Nearest Neighbor heuristic for the hypertour problem**

Set Hypertour $T = \{\}$,
No component types are assigned to feeder slots yet,
Select arbitrary starting placement location $p^{start}$,
Set $p_0 = p^{start}$,
**While** not all placement locations visited **Do**
- set $A = \{p_0\}$,
- select nearest feeder slot to $p_0$, $s_0$, whose components have not all been placed yet,
- **If** no component type assigned to $s_0$ yet
  - ○ select nearest placement location $p$ to $s_0$ whose component type has not been assigned yet,
  - ○ assign component type of $p$ to $s_0$,
- **Else**
  - ○ select nearest empty placement location $p$ to $s_0$ of the same component type,
- **End If**
- add $p$ to $A$,
- **While** not all placement locations visited **and** still head available **Do**
  - ○ select to $s_0$ nearest feeder slot $s^{next}$ whose components have not all been placed yet,
  - **If** no component type assigned to $s^{next}$ yet
    - ◇ select nearest placement location $p^{next}$ to $p$ whose component type has not been assigned yet,
    - ◇ assign component type of $p^{next}$ to $s^{next}$,
  - **Else**
    - ◇ select nearest empty placement location to $p$, $p^{next}$ of same component type as $s^{next}$,
  - **End If**
  - ○ add $p^{next}$ to $A$,
  - ○ set $p = p^{next}$,
- **End While**
- Calculate "weighted" sequence length for all feasible subsequences of $A$,
- Select subsequence of minimal "weighted" length $A^*$,
- Add $A^*$ as a hyperedge to hypertour $T$,
- Choose the last element of $A^*$ as the new starting position $p_0$,

**End While**
**Return** Hypertour $T$.

Starting at a placement location, instead of looking for the nearest feeder slot and its corresponding placement locations which still have to be fed, when adding a placement location to $A$ we compare the distances to all placement locations that have not been visited yet. The overall schema works then the same way as described above. The time complexity of this approach to choose the next placement location is $O(f \cdot n)$, as starting at a placement location we compare the distances to $n$ placement locations for all $f$ feeder slot. Again this process is repeated $n$ times leading to a time complexity of $O(f \cdot n^2)$. This approach is with its time complexity of $O(f \cdot n^2)$ more expensive than the approach outlined before. However, it gives access to a larger neighborhood which may improve the quality of the tour constructed.

We will now consider a local improvement algorithms for the hypertour problem based on the local search algorithm $k$-opt for the TSP [8], [11]. One possibility is to directly apply $k$-opt as defined for the TSP. This algorithm works for the hypertour problem subject to the restriction that each hyperedge has maximal length of $h + 1$. The algorithm consists of applying simple tour modifications to a feasible tour. This process is then performed repeatedly as long as improved tours are found. This way the neighborhood of a feasible tour is searched in order to reach a local minimum.

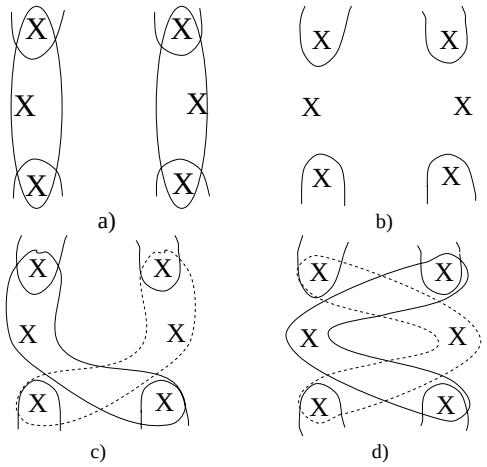


**Figure 8. Schema of hypertour $2$-opt moves**

In our $k$-opt approach we remove and attach $k$ hyperedges rather than $k$ single edges as for the TSP. First, $k$ hyperedges are removed, second we create new hyperedges by selecting optimal feasible sequences reconnecting the "free" elements of the removed hyperedges with the rest of the tour. We illustrate this procedure for a hypertour 2-opt moves in figure 8, where fig. 8(a) illustrates the initial 2 hyperedges, fig. 8(b) show the "free" elements after removing the 2 hyperedges and fig. 8(c) and (d) two possible moves. This introduces a larger neighborhood and more possible moves than TSP $k$-opt and also allows moves where the fixed start and end locations of a hyperedge stay the same, as shown in figure 8(d). The neighborhood sizes for 2-opt moves considering a 2,3 or 4 headed placement machine are given in table 3. We can see that the time complexity may be prohibitive for large problems.

**Table 3. Neighborhood sizes for $2$-opt hypertour moves**

| Number of "free" elements | Number of heads 2 | 3 | 4 |
|---|---|---|---|
| 1 | 4 | 4 | 4 |
| 2 | 4 | 12 | 12 |
| 3 | – | 24 | 48 |
| 4 | – | 48 | 144 |
| 5 | – | – | 480 |
| 6 | – | – | 1440 |

A drawback of $k$-opt for hypertours is that it only modifies the placement sequence but not the assignment of component types to feeder slots. In order to overcome this problem we suggest that two or more feeder slots may also form part of a $k$-opt move and they will then be rearranged at the same time as the hyperedges.

With the hypertour construction heuristic and the local search algorithm described in this section, good locally optimal solutions can be determined for the hypertour problem. Allowing the $k$-opt for hypertours algorithm to accept non-improving tours enables the use of meta-heuristics as *tabu search* [8], *simulated annealing* [8] or *variable neighborhood search* [6] to overcome the problem of local minima.

## 5. Conclusion

This paper has presented a family of placement machines used in printed circuit board assembly along with models for some general and machine specific problem characteristics arising in practice. We have presented a new model for multi-headed placement machines. For this problem a suitable sequence construction algorithm and a local search method to quickly find good local optima have been proposed.

In the future we are planning to implement the algorithms described in section 4 along with tabu search, simulated annealing and variable neighborhood search and apply them to random and industrial data. It will be interesting to see whether increasing the neighborhood size, by increasing

$k$ or the use of meta-heuristic techniques is more effective for the hypertour problem.

## References


[1] J. Ahmadi, S. Grotzinger, and D. Johnson. Component allocation and partitioning for a dual delivery placement machine. *Operations Research*, 36(2):176–191, 1988.

[2] M. Ball and M. Magazine. Sequencing of insertion in printed circuit board assembly. *Operations Research*, 36(2):192–210, 1989.

[3] J. Bard, R. Clayton, and T. Feo. Machine setup and component placement in printed circuit board assembly. *International Journal of Flexible Manufacturing Systems*, 6(1):5–31, 1994.

[4] C. Chen. Planning optimal precedence-constraint robot assembly sequences problem with neural computation. In *Applications of Artificial Intelligence VIII*, pages 320–331, 1990.

[5] L. Foulds and H. Hamacher. Optimal bin location and sequencing in printed circuit board assembly. *European Journal of Operational Research*, 66(3):279–290, 1993.

[6] P. Hansen and N. Mladenovic. Variable neighborhood search: Principles and applications. In *Invited papers at Euro XVI*, 1998.

[7] J. Hong, S. Lee, and B. Lee. Hierarchical optimization method in the PCB assembly for surface mounting machines. In *Proceedings of the 1997 IEEE International Symposium on Industrial Electronics, ISIE*, pages 129–134, 1997.

[8] D. Johnson and L. McGeoch. The Travelling Salesman Problem: A case study. In *Local Search in Combinatorical Optimization*. Wiley, 1997.

[9] N. Khoo and T. Ng. A genetic algorithm-based planning system for PCB component placement. *Int. J. of Prod. Economics*, 54:321–332, 1998.

[10] R. Kumar and H. Li. Integer programming approach to printed circuit board assembly time optimization. *IEEE Transactions on Components, Packaging, and Manufacturing Technology Part B: Advanced Packaging*, 18(4):720–727, 1995.

[11] E. Lawler, J. Lenstra, A. R. Kan, and D. Shmoys. *The Travelling Salesman Problem: A guided tour of combinatorial optimization*. Wiley, 1985.

[12] S. Lee, J. Hong, D. Kim, and B. Lee. Effective algorithm for a surface mounting machine in printed circuit board assembly. In *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems*, pages 932–937, 1997.

[13] T. Leipala and O. Nevalainen. Optimization of the movements of a component placement machine. *European Journal of Operational Research*, 38(2):167–177, 1989.

[14] V. Leon and B. Peters. Replanning and analysis of partial setup strategies in printed circuit board assembly systems. *International Journal of Flexible Manufacturing Systems*, 8(4):289–411, 1996.

[15] L. Moyer and S. Gupta. An efficient assembly sequencing heuristic for printed circuit board configurations. *Journal of Electronics Manufacturing*, 7(2):143–160, 1997.

[16] M. Ng. Heuristics approach to printed circuit board insertion problem. *Journal of the Operational Research Society*, 49(10):1051–1059, 1998.

[17] J. Park and H. Asada. Sequence optimization for high speed robotic assembly using simulated annealing. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3441–3446, 1994.

[18] S. Shevell, J. Buzacott, and M. Magazine. Simulation and analysis of a circuit board manufacturing facility. *Proceeding of the 1986 Winter Simulation Conference (December)*, Washington, D.C, 1986.