No. 2010-128

A MULTI-OBJECTIVE OPTIMIZATION APPROACH FOR MULTI-HEAD BEAM-TYPE PLACEMENT MACHINES

By S.A. Torabi, M. Hamedi, J. Ashayeri

December 2010

ISSN 0924-7815

A Multi-Objective Optimization Approach for Multi-Head Beam-Type Placement Machines

S.A. Torabi [a], M. Hamedi [a] and J. Ashayeri [b]

[a] Department of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran

satorabi@ut.ac.ir, mhamedy@ut.ac.ir

[b] Department of Econometrics & Operations Research, Tilburg University, P.O. Box 90153, LE Tilburg, The Netherlands

j.ashayeri@uvt.nl

JEL Code: C6, M110

Abstract This paper addresses a highly challenging scheduling problem in the field of printed circuit board (PCB) assembly systems using Surface Mounting Devices (SMD). After describing some challenging optimization sub-problems relating to the heads of multi-head surface mounting placement machines, we formulate an integrated multi-objective mathematical model considering of two main sub-problems simultaneously. The proposed model is a mixed integer nonlinear programming one which is very complex to be solved optimally. Therefore, it is first converted into a linearized model and then solved using an efficient multi-objective approach, i.e., the augmented epsilon constraint method. An illustrative example is also provided to show the usefulness and applicability of the proposed model and solution method.

Keywords PCB assembly . Multi-head beam-type placement machine . Multi-objective mathematical programming . Augmented epsilon-constraint method

1 Introduction

Over the last two decades, the assembly of PCBs has generated a huge amount of industrial activity. One of the major developments in PCB assembly is the introduction of surface mount technology (SMT) in 1960s. SMT has displaced through-hole technology as the primary means of assembling PCBs. It has also made it easy to automate the PCB assembly process. The component placement machine is probably the most important piece of manufacturing equipment on a surface mount assembly line [1]. As SMT becomes popular, different types of placement machines have arisen. For a well-organized classification of placement machines based on their operational methods the reader is referred to Ayob and Kendall [2]. Among the component placement machines, multi-head of gantry-type machines are becoming increasingly popular because they provide high mounting speed with relatively low cost. A gantry robot, which moves components between the components feeder racks and the PCB, usually involves multiple heads to reduce the number of pick-and-place cycles. The heads are sequentially arranged on a beam or a rotating wheel at the gantry robot. The former is called beam-type while the latter is called collect-and-place type [3]. Both types of these machines can have single or multiple arms. The proper assignment of component types to feeders in placement machines and the placement sequence of components on the PCB are the main factors greatly affecting the production cycle time of each machine and the whole SMT line [4]. These problems are highly interrelated and very difficult to solve simultaneously. Therefore, during the last decade, most research on minimizing the PCB assembly time has focused on solving these problems separately by decoupling one from the other [5]. Many research works have been devoted to these complex problems by developing various mathematical models and solution approaches. For example, Ball and Magazine [6] modeled the sequencing problem as a directed postman problem. They suggested that the balance and connect heuristic can be applied to this problem. Leipala and Nevalainen [7] dealt with the placement sequencing sub-problem as a three dimensional asymmetric travelling salesman problem whilst the feeder assignment sub-problem was modeled as a quadratic assignment problem. Or and Duman [8] used a convex hull algorithm and Or-opt tour improvement method for placement sequencing and feeder assignment sub-problems. Khoo and Loh [9] modeled the problem of assembling a printed circuit board with a chip shooter as a multi-objective problem. They applied a genetic algorithm to generate the placement sequences and feeder assignment. Ho and Ji [10] developed a hybrid genetic algorithm to integrate placement sequencing, feeder assignment and component retrieval sub-problems. Their purposed algorithm was found to perform better than conventional genetic algorithms.

Moon [11] developed two different methods using special features on printed circuit boards to simultaneously improve component's rack assignment and component mounting sequencing problems in chip shooter machines like Panasert MSH-II, Fuji CP-II, and CP-IV. Based on results from field surveys, it is found that identical components are positioned closely with each other or identical single boards are repeatedly printed on one big board to enlarge up to a proper size to be assembled in the machine. These patterns are adapted on the design of assembly methods to increase productivity. Simulation models are also constructed for performance evaluation purposes of the developed heuristics.

SMD machines with multiple heads are the most popular ones in SMT lines, but the complexity of their performance makes the respective optimization problems more difficult to be solved. However, the literature review regarding these machines is rather scarce. Van Laarhoven and Zijim [12] applied a hierarchical procedure for solving the optimization problems of a set of beam-type multi-head placement machines with three placement heads. All sub-problems in the hierarchy were solved sequentially by simulated annealing approach. They stated that their proposed method performs well in balancing the workload over the machines. Magyar et al. [13] dealt with the problem of sequencing of pick-and-place cycles; allocation of nozzles to heads; and feeder assignment using a hierarchical approach. They considered a general surface mounting (GSM) machine that is a beam-type multi-head placement machine. Initially, they solved the feeder assignment sub-problem by using a greedy local search. The output of first sub-problem is used as the input for nozzle optimization sub-problem and the output of nozzle optimization sub-problem considered as an input to component pick-and-place sub-problem that is also solved using a greedy local search approach. Their approach significantly decreased the cycle time. Lee et al. [5] applied a genetic algorithm for a joint-solution of the optimization sub-problems in a multi-head beam-type placement machine. They converted the optimization problem of a multi-head machine to a single-head case by grouping feeders and clustering of components. They utilized single-head methods to the multi-head case. They also selected the partial-link structure for the chromosomes. Hong et al. [14] implemented a biological immune algorithm for optimization problem of a multi-head beam-type placement machine. Jeevan et al. [15] applied a genetic algorithm to minimize the cycle time in a beam-type multi-head machine. They used the distance of a TSP tour as the fitness function of genetic algorithm. However, they did not discuss the mathematical modeling and chromosome definition in the paper. Grunow et al. [16] followed a hierarchical approach for optimization problem of a collect-and-place multi-head placement machine. They considered four sub-problems in their proposed hierarchy, i.e., (i) feeder assignment; (ii) sub-tours composition; (iii) sequencing of placement of components within a sub-tour; (iv) sequencing the sub-tours. A three-stage approach is applied for solving the sub-problems. Sub-problem (i) is solved in stage one using a greedy algorithm. In the second stage sub-problems (ii), (iii) and (iv) are solved by modeling them as a vehicle-routing problem. Given the feeder assignment solution from stage one, the authors sequence the component pick-and-place operations using a heuristic approach. The final stage of solution approach improves the feeder assignment and the component pick-and-place sequence using a random descent 2-opt swapping procedure. Sun et al. [17] considered the optimization performance of a dual-gantry collect-and-place multi-head placement machine. They proposed a hybrid genetic algorithm for solving the component allocation and feeder assignment sub-problems along with a greedy algorithm for placement heads workload balancing. Raduly-Baka and Knuutila [18] presented different approaches for determining the number of nozzles for populating a PCB type by a multi-head beam-type machine. They assumed that each component type can be handled using one nozzle type. Their nozzle selection problem optimally solved using a three-phase greedy algorithm. They also investigated the nozzle selection in the case of multiple PCB types. Kulak et al. [19] proposed three different genetic algorithms for scheduling operations of a collect-and-place placement machine. They considered the case of single and dual-gantry placement machines. They integrated feeder assignment and placement sequencing using a genetic algorithm. The authors claimed that their proposed genetic algorithms are very efficient in terms of computational time, especially if adequate coding schemes are used. Recently, Sun and Lee [3] developed a branch-and-price procedure for a placement routing problem for a beam-type multi-head placement machine. They formulated the problem as an integer

programming model with a huge number of variables. They solved the linear relaxation of the model by a column generation method. Li et al. [20] considered the cycle time minimization of a SONY SE-1000 machine which belongs to collect-and-place multi-head placement machines. They assumed that the mounting sequence is given in advance and they solved feeder assignment sub-problem using a genetic algorithm. In their proposed genetic algorithm, a uniform order crossover and exchanging mutation is applied.

Literature review regarding the multi-head SMD placement machines reveals less attention to optimal utilization of the placement heads. In addition to this gap, the dependency of the moving speed of the robotic arm to the combination of nozzles and components currently loaded on the heads is also neglected. Our focus in this paper is on single arm beam-type multi-head placement machines. In this regard, we develop a novel mathematical model to deal with the heads-related decision problems in such placement machines by addressing the existing gaps in the literature.

The remainder of this paper is organized as follows. Section 2 contains a more detailed description of a single arm beam-type multi-head placement machine. Section 3 provides a precise statement of the problem and its sub-problems, and presents a new integrated model for the main sub-problems of heads, i.e., the workload balancing and nozzle selection simultaneously. The solution procedure is elaborated in Section 4. An illustrative example is provided in Section 5. Finally, concluding remarks are given in Section 6.

2 Machine description

Fig. 1 illustrates the schematic view of the considered multi-head beam-type placement machine in this paper. SMD machines such as Yamaha YV-64/88/100, Samsung CP-40/50 and Juki KE-750/760 belong to this type of placement machines. The machine has a fixed PCB table, a feeder bank, an arm that is equipped with a number of placement heads and an Automatic Nozzle Changer (ANC). The PCB remains fixed on the PCB table during the placement process. A fixed feeder bank is located on one side of the PCB table. The feeder bank consists of a number of slots for positioning the feeders. Electronic Components are supplied to the machine by feeders. Multiple heads are located on the arm and move together with it simultaneously in both X and Y directions. The assembly process starts by moving the arm toward the feeder bank and picking up at most H components either simultaneously or on one by one basis by moving along the feeder bank. Then it moves to the PCB to place the components just picked up on the specific locations on the PCB. When the head positions exactly on the placement location, it moves down in Z direction and mounts its component on the board. Each head can use various types of nozzles for picking and placing components. Not each nozzle is suitable for handling each component type. Large nozzles cannot pick small components and small nozzles cannot pick large components. Therefore, it will be necessary to exchange nozzles sometimes. The nozzles are stored in ANC (automatic nozzle changer). The exchange action starts by moving the arm to the ANC and inserting the unnecessary nozzle in an empty slot. Then, the arm moves towards the new nozzle and picks it. Notably, this nozzle exchange process is often time-consuming which should be avoided as many as possible.

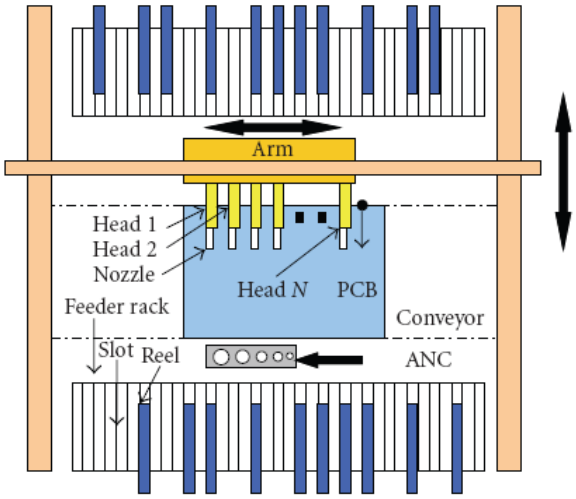

Fig. 1 The schematic view of a multi-head machine.

3 Problem definition and formulation

3.1 Problem hierarchy

Given a PCB type to be mounted with N components divided into T types using a multi-head beam-type placement machine equipped with H placement heads,

the main problem can be described through the following sub-problems:

1) Assignment of feeders to feeder slots.
2) Partitioning the *N* components into a number of clusters, each of which consisting of at most *H* components (pertaining to a pick-and-place tour).
3) Sequencing of component clusters and within each cluster, placement sequencing of its components, so that the cycle time of the machine (i.e., the necessary time to mount all components on the PCB) is minimized.

Obviously this problem is extremely complex. In order to reduce this complexity, a hierarchical decomposition approach is often applied (See for example Ayob and Kendall [21]). In hierarchical approach, the main problem is decomposed into a series of sub-problems in such a way the solution of each sub-problem generates required input data for the next sub-problem. The following sub-problems are a more detailed description of above-mentioned sub-problems:

i. For each pair of component type *t (t=1,...,T)* and head *h* (*h=1,...,H)*, determine the number of components of type *t* to be handled by head *h*.
ii. For each pair of component type and head, determine the most appropriate nozzle type handling the component.
iii. For each pair of component type and head, determine which components of type *t* are to be mounted by head *h*.
iv. Determine the component clusters.
v. Sequence the component clusters.
vi. Sequence the components placing within each cluster.

3.2 Problem statement and assumptions

This problem was inspired by a real case in Assembléon during a consultative work done by one of the authors with the company. Assembléon which was formerly known as Philips Electronic Manufacturing Technology, develops, assembles, and distributes a diverse range of SMD machines (especially, single arm beam-type multi-head placement machines) and provides a broad range of related services. In this paper, the best way of distribution (assignment) of components over the heads of a single arm beam-type multi-head placement machine is considered as a separate objective in the proposed mathematical model to minimize the load of bottleneck head (i.e., the head with maximum load among others). Furthermore, in order to handle the components on the heads, vacuum nozzles are applied for pick and place operations. Throughout the literature of SMD machines, the compatibility degree of each pair of nozzle-component type has always been considered in a 0-1 manner i.e., it is assumed that a nozzle is capable of handling a component type or not. But in the real world the story is completely different, i.e., each component type can be handled by different nozzle types with different degrees of compatibility. Therefore, for the first time in the literature of SMD machines, we introduce the appropriateness factors to evaluate the compatibility of each pair of nozzle-component type. Practically, we should try to choose the most suitable nozzles for handling the components on the heads because the speed of the robotic arm depends on the combination of nozzles and components currently loaded on the heads. That is, large components picked with a small nozzle cannot be moved as fast as smaller components picked with the same nozzle. However, if the most suitable nozzles are applied for handling the components, the arm can move faster. Accordingly, maximizing the nozzles' appropriateness function as the summation of all corresponding appropriateness factors in handling the components is introduced as the second objective. Notably, these two objective functions are partially conflicting objectives, i.e., in the most cases (not always) adopting the best nozzles for handling the components on the heads may result in unwanted nozzle exchanges which directly affects the workload of the heads. Therefore, finding a trade-off between the workload of bottleneck head and the total appropriateness is of particular interest. It should be noted that since the number of nozzle exchanges affects the workload of the heads directly, it is important to recognize that the minimization of workload of bottleneck head does not necessarily imply the minimization of the diversity of nozzle types.

Now we formulate the main sub-problems affecting the performance of utilizing the heads greatly, i.e., the sub-problems i and ii of aforementioned problem hierarchy together as an integrated model in such a way that:

1. The number of nozzle exchanges is minimized.

2. Each component type is handled by the most appropriate nozzle.
3. The machine heads are loaded with the approximately same workloads. In other words, the numbers of components which are assigned to each head; are approximately equated through minimizing the load of bottleneck head.

The assumptions made in formulating the concerned sub-problems are as follows:

- There is one feeder rack located in one side of the PCB table.
- The number of heads is given in advance.
- Each component type must be handled by exactly one nozzle on each head.
- The order, in which the heads place the components at each pick and place tour is given, i.e., the first head places its component first, then the second head places its component, and so on.
- There are multiple copies of each nozzle. Each nozzle is automatically changed at the *automatic nozzle changer* (ANC) when it cannot grip the required component.
- The compatibility of each pair of component-nozzle is evaluated by the appropriateness factors. We consider λ_{tq} as appropriateness factor (degree) when nozzle q handles a component of type t. These factors can be considered as fuzzy or crisp numbers. But sufficiently, here we assume that they are crisp numbers, i.e., 0, 1, 3, 5, 7, 9 where zero is considered for the case that a nozzle cannot manipulate a component type. Other factors, i.e., 1, 3, 5, 7, 9, denote the very low, low, medium, good and very good appropriateness degrees, respectively.
- We may confront with a case that some component types cannot be handled by available nozzles. In such a case, we ignore this component type and it is manipulated at the next stage manually.

3.3 Problem formulation

The following notations are used in the model formulation:

Indices:

Index of component types $t = 1, 2, \dots, T,$

Index of nozzles $q = 1, 2, \dots, Q,$

Index of heads $h = 1, 2, \dots, H,$

Index of components $i = 1, 2, \dots, N.$

Parameters:

γ_h Constant cost (time) of exchanging a nozzle on head h

λ_{tq} The appropriateness factor when nozzle q handles components of type t

N_t Total number of components of type t; $(\sum_{t=1}^{T} N_t = N)$

d_t The average distance of components of type t on the PCB from the center of feeder rack

$\bar{v}$ Average velocity of the robotic arm motion

f_t The total time to pick a component of type t when the head is positioned above the feeder plus the time to place the component when the head is exactly positioned above the PCB.

Variables

x_{th} Total number of components of type t that are assigned to head h

z_{tqh} $\begin{cases} 1\ ; \text{ if component type } t \text{ is handeled} \\ \quad \text{by nozzle } q \text{ on head } h \\ 0\ ; \text{ otherwise} \end{cases}$

S_{qh} $\begin{cases} 1\ ; \text{ if nozzle } q \text{ is assigned to head } h \\ 0\ ; \text{ otherwise} \end{cases}$

μ_h Total number of nozzle exchanges on head h (which its maximum value is equal to the number of components assigned to head h minus 1)

Using the aforementioned notations, the mathematical formulation of the problem can be written as follows:

$$min\left\{max_h\left(\gamma_h\mu_h + \sum_{t=1}^{T}\left(\frac{2d_t}{\bar{v}} + f_t\right).x_{th}\right)\right\} \quad (1)$$

$$max \sum_{h=1}^{H}\sum_{q=1}^{Q}\sum_{t=1}^{T}\lambda_{tq}.z_{tqh} \quad (2)$$

Subject to:

$$\mu_h \geq \sum_{q=1}^{Q} S_{qh} - 1 \qquad h = 1, \dots, H \quad (3)$$

$$\sum_{t=1}^{T} z_{tqh} \leq T.S_{qh} \qquad h = 1, \dots, H; q = 1, \dots, Q \quad (4)$$

$$\sum_{t=1}^{T} z_{tqh} \geq S_{qh} \qquad h = 1, \dots, H; q = 1, \dots, Q \quad (5)$$

$$x_{th} \leq N_t.\sum_{q=1}^{Q} z_{tqh} \qquad h = 1, \dots, H; t = 1, \dots, T \quad (6)$$

$$x_{th} \geq \sum_{q=1}^{Q} z_{tqh} \qquad h = 1, \dots, H; t = 1, \dots, T \quad (7)$$

$$\sum_{q=1}^{Q} z_{tqh} \leq 1 \qquad h = 1, \dots, H; t = 1, \dots, T \quad (8)$$

$$\sum_{h=1}^{H} x_{th} = N_t \qquad t = 1, \dots, T \quad (9)$$

$$x_{th} \geq 0 \; and \; Integer \qquad h = 1, \dots, H; t = 1, \dots, T \quad (10)$$

$$\mu_h \geq 0 \; and \; Integer \qquad h = 1, \dots, H \quad (11)$$

$$z_{tqh} \in \{0,1\} \qquad h = 1, \dots, H; t = 1, \dots, T; q = 1, \dots, Q \quad (12)$$

$$S_{qh} \in \{0,1\} \qquad h = 1, \dots, H; q = 1, \dots, Q \quad (13)$$

The objective function (1) indicates that the workload of bottleneck head is minimized. The workload of each head consists of two terms. The first term is the necessary time for nozzle exchanges and the second one is an estimation of the time that the head travels above the PCB and feeder rack. It is noteworthy that since the feeder assignment and placement sequencing sub-problems will be solved after the concerned ones here, the exact moving path of the robotic arm is not given at this time. Therefore, an estimation of the real traveling time is used for calculating the workload of a head by considering the average distance of the locations of a component type (for all component types) on the PCB from the center of feeder rack. The second objective tries to maximize the appropriateness function of using suitable nozzles for components. Constraints (3) express the relation between the number of nozzles and components that are assigned to head *h* and the number of nozzle exchanges. The following expressions explain why constraints (4) and (5) have been introduced:

$$\begin{cases} if\ \sum_{t=1}^{T} z_{tqh} = 0\ \ then\ S_{qh} = 0 \\ if\ \sum_{t=1}^{T} z_{tqh} > 0\ \ then\ S_{qh} = 1 \end{cases}$$

$$\Rightarrow \begin{cases} \sum_{t=1}^{T} z_{tqh} \le M.S_{qh} \\ \sum_{t=1}^{T} z_{tqh} \ge S_{qh} \end{cases} \quad for\ all\ q\ and\ h$$

Notably, when all of assigned component types to a head can be handled using one nozzle type; the maximum value of *M* is equal to *T*; hence *M* has been replaced with *T* in constraint (4).

Constraints (6) and (7) state that when a component type is assigned to a head; only one nozzle must be selected to handle it. The following expressions describe how they have been formulated:

$$\begin{cases} if\ x_{th} = 0\ then\ \sum_{q=1}^{Q} z_{tqh} = 0 \\ if\ x_{th} > 0\ then\ \sum_{q=1}^{Q} z_{tqh} = 1 \end{cases}$$

$$\Rightarrow \begin{cases} x_{th} \le M'.\sum_{q=1}^{Q} z_{tqh} \\ x_{th} \ge \sum_{q=1}^{Q} z_{tqh} \end{cases} \quad for\ all\ t\ and\ h$$

If all components of type *t* are allocated to head *h*, the maximum value of M' can be replaced by N_t. Equation (8) ensures that if a component type is assigned to a head, it must be handled by only one nozzle. Constraint (9) guarantees the dispersion of all component types among the heads. Constraints (10) and (11) show the integrality and non-negativity of variables x_{th} and μ_h. Finally, constraints (12) and (13) show that z_{tqh} and S_{qh} variables are binary.

3.3 Linearization

The first objective could simply be linearized as follows:

Let $max_h \left(\gamma_h \mu_h + \sum_{t=1}^{T} \left(\frac{2d_t}{\bar{v}} + f_t\right).x_{th}\right) = \beta$

Hence the objective (1) can be modified to: min β; with adding the following constraints to the model:

$$\beta \ge \gamma_h \mu_h + \sum_{t=1}^{T} \left(\frac{2d_t}{\bar{v}} + f_t\right).x_{th}; \quad for\ h = 1, \dots, H$$

Therefore, the linearized model can be written as model (3)-(16).

4 Solution procedure

4.1 An overview of Multi-Objective Programming

A general multi-objective optimization problem consists of a number of objectives to be optimized simultaneously in the feasible region. The general formulation of multi-objective optimization problems can be written in the following form:

$$Max(f_1(\boldsymbol{x}), \dots, f_p(\boldsymbol{x}))$$
$$subject\ to:\ \ \boldsymbol{x} \in S \qquad (17)$$

In this formulation: $f_i(\boldsymbol{x})$ denotes the *i*th objective function and S indicates the feasible space. The ultimate goal is simultaneous maximization of given objective functions. When, as in most cases, some of the objective functions conflict with each other, there is no exactly one solution but many alternative solutions. Such potential solutions which cannot improve all the objective functions simultaneously are called efficient (Pareto optimal) solutions.

$$min\,\beta \tag{14}$$

$$max \sum_{h=1}^{H}\sum_{q=1}^{Q}\sum_{t=1}^{T} \lambda_{tq} \cdot z_{tqh} \tag{15}$$

Subject to:

$$\beta \geq \gamma_h \mu_h + \sum_{t=1}^{T} \left(\frac{2d_t}{\bar{v}} + f_t\right) . x_{th};\ h = 1, \dots, H \tag{16}$$

Constraints (3)-(13).

A feasible solution *x* is called efficient if there does not exist another feasible solution say x' such that $f_i(x') \geq f_i(x)$ for all values of *i* with at least one strict inequality. In other words, a solution x is called Pareto optimal if there is no other $x' \neq x$ that increases some objective functions without degrading at least one other objective function. Under this definition, we usually find several efficient solutions estimating the trade-off surface. In this sense, the search for an optimal solution has fundamentally changed from what we see in the case of single-objective problems. However, users practically need only one solution from the set of efficient solutions. According to Miettinen [22], the multi-objective solution approaches can be classified into the four categories based on the phase in which the decision maker involves in the decision making process: The first one does not use any preference information (called no-preference). These methods solve a problem and give a solution directly to the decision maker. The second one is to find all possible efficient solutions and then using the decision maker's preferences to determine the most suitable one (called posteriori methods). The third approach is to incorporate the preference information before the optimization process often in terms of objectives' weights resulting in only one solution at the end (called priori methods). The fourth approach (called interactive methods) is to hybridize the second and third ones in which the decision maker's preferences is periodically used to refine the obtained efficient solutions leading to guide the search space more efficiently. In general, the second one, i.e., the posteriori approach is mostly preferred by the researchers and practitioners since it is less subjective than the others. By using posteriori methods, the decision maker is provided by a set of Pareto optimal solutions and the most suitable one is finally selected based on her/his preferences. Here, the two most popular posteriori methods, i.e., the weighted sum and ε-constraint methods are described briefly.

In the weighted-sum method, all the objectives are aggregated into a single objective by using a weight vector. Although the weighted-sum method is simple and easy to use, there are two major problems. Firstly, there is the difficulty of selecting the weights in order to deal with scaling problems since the objectives usually have different magnitudes causing biases when

searching for trade-off solutions. Secondly, the performance of the method is heavily dependent on the shape of the Pareto optimal frontier so that it cannot find all the optimal solutions for problems that have a non-convex Pareto optimal frontier. To overcome these difficulties, the ε-constraint method has been introduced in which only one objective is optimized while the others are moved to constraints. The ε-constraint method for solving model (17) can be shown as follows:

$$
\begin{aligned}
&max\ f_1(\boldsymbol{x}) \\
&\text{s.t.} \\
&f_2(\boldsymbol{x}) \geq e_2, \\
&f_3(\boldsymbol{x}) \geq e_3, \qquad (18) \\
&\dots \\
&f_p(\boldsymbol{x}) \geq e_p, \\
&\boldsymbol{x} \in S.
\end{aligned}
$$

By this method, via systematic variation in the RHS of the constrained objective functions (i.e., the e_i values) and solving the respective single-objective models, the efficient solutions can be obtained effectively. Although the ε-constraint method does not suffer from the difficulties that the weighted-sum does, some ambiguities about this method are considerable. In order to resolve these ambiguities, recently, Mavrotas [23] proposes a novel version of the conventional ε-constraint method, i.e., the augmented ε-constraint method (hereafter it is called AUGMECON) which is discussed in more details at below.

4.2 The proposed solution method

To find the most preferred efficient solution of the proposed bi-objective model, we apply the augmented ε-constraint method. Although ε-constraint method has several advantages over the other posteriori methods, three points about the implementation of this method should be taken into account:

a. The estimation of the range of objective functions over the efficient set
b. The guarantee of efficiency of the obtained solutions
c. The increased solution time for problems with more than two objectives.

In order to tackle these issues, Mavrotas [23] presents a novel version of the conventional ε-constraint method, i.e., the augmented ε-constraint (AUGMECON) method. Here we take a closer look at this method to see how it can be implemented in practice. The first step in applying the ε-constraint method is to determine the range of objective functions which are used as constraints. To do so, we should calculate the best (ideal) and worst (nadir) values of objective functions over the feasible space. The best value could be calculated as the optimal solution of individual optimization over the feasible space but the worst value is not easily attainable. Usually, the worst value is estimated from the payoff table (a table which is comprised of the results of individual optimization of objective functions). In this manner, the worst value of each objective function is approximated with selecting the minimum value of corresponding column.

In the case of alternative optima, the solutions obtained from the individual optimization of objective functions may not be an efficient but weakly efficient one. In order to overcome this ambiguity, Mavrotas [23] proposes the use of lexicographic optimization for each objective function to construct the payoff table ensuring to yield just Pareto optimal solutions. The lexicographic optimization is applied as follows. We optimize the first objective function, obtaining $\max f_1 = z_1^*$. Then, we optimize the second objective function by adding the constraint $f_1 \geq z_1^*$ in order to keep the

optimal solution of the first optimization. Assume that we obtain max$f_2 = z_2^*$. Subsequently, we optimize the third objective function by adding the constraints $f_1 \geq z_1^*$ and $f_2 \geq z_2^*$ in order to keep the previous optimal solutions and so on, until we finish with the objective functions.

The second point is that the optimal solution of the conventional ε-constraint is guaranteed to be an efficient solution only if all the (*p-1*) objective functions' constraints are biding; otherwise, if there are alternative optima (that may improve at least one of the non-binding constraints that corresponds to an objective function), the obtained optimal solution of the problem is not in fact efficient, but is a *weakly* efficient solution [24]. In order to overcome this ambiguity Mavrotas [23] proposes the transformation of the objective function constraints to equalities by introducing slack or surplus variables. In the same time, these slack or surplus variables are used as a second term (with lower priority) in the objective function to force the model to produce only efficient solutions. In this way, the new problem (AUGMECON model) can be written as follows:

$$max(f_1(x) + eps \times (s_2 + s_3 + \cdots + s_p))$$
s.t.
$$f_2(x) - s_2 = e_2$$
$$f_3(x) - s_3 = e_3 \quad (19)$$
$$f_p(x) - s_p = e_p$$
$$x \in S \; and \; s_i \in R^+$$

Where *eps* is a small number (usually between 10^{-6} and 10^{-3}). Mavrotas [23] proves that AUGMECON produces only efficient solutions i.e., it avoids to generate weakly efficient solutions. In order to avoid any scaling problems Mavrotas [23] recommends to replace the s_i in the second term of the objective function by s_i/r_i, where r_i is the range of *i*th objective function obtained from payoff table. Thus, the final version of augmented ε-constraint method is written as follow.

$$max(f_1(x) + eps \times (s_2/r_2 + s_3/r_3 + \cdots + s_p/r_p))$$
s.t.
$$f_2(x) - s_2 = e_2$$
$$f_3(x) - s_3 = e_3 \quad (20)$$
$$\cdots$$
$$f_p(x) - s_p = e_p$$
$$x \in S \; and \; s_i \in R^+$$

The third point in the conventional ε-constraint method is the additional computations when the problem becomes infeasible. Mavrotas [23] adds an innovative addition to the algorithm, i.e., the early exit from the nested loops when the problem becomes infeasible. He state that this issue can accelerate the algorithm speed significantly in the case of having several (more than two) objective functions.

Practically, the AUGMECON method is implemented as follows: From the payoff table we obtain the range of each (*p*-1) objective functions that are going to be used as constraints. Then we divide the range of the *i*-th objective function into q_i equal intervals using (q_i-1) intermediate equidistant grid points. Thus, we obtain in total (q_i+1) grid points that are used to vary parametrically the RHS of the *i*-th objective function (e_i). Therefore, the total number of single-objective models (runs) which certainly lead to the generation of efficient solutions becomes (q_2+1)×(q_3+1)×⋯×(q_p+1) ones.

4 An illustrative example

In this section to show the applicability and usefulness of the proposed model and solution method, we provide an illustrative example, for which we generate a set of 5 efficient solutions using the augmented epsilon constraint method. The inputs of the sample problem are summarized through Tables 1-4.

Table 1 The parameters of sample problem

Number of	heads	component types	nozzles
Value	3	10	4

Table 2 The values of *d(t)*, *f(t)* and *N(t)*

t	1	2	3	4	5	6	7	8	9	10
d(t)	7	6	4	7	5	4	4	7	3	2
f(t)	0.1	0.2	0.1	0.5	0.2	0.3	0.3	0.9	0.9	0.8
N(t)	3	4	6	2	2	2	3	4	9	7

Table 3 The values of γ_n

Head	1	2	3
$\boldsymbol{\gamma_n}$	1	1	1

Table 4 Values of $\boldsymbol{\lambda(t,q)}$

		q			
		1	2	3	4
	1	1	5	2	5
	2	1	4	3	5
	3	4	3	7	5
	4	4	6	6	5
t	5	2	5	4	4
	6	3	7	1	5
	7	1	6	4	2
	8	4	4	3	2
	9	3	2	5	1
	10	5	5	6	2

The first step in applying the AUGMECON is to construct the payoff table using the lexicographic optimization as follows:

First we optimize the first objective function over the feasible region by which we obtain the optimal solution (37.57, 0) in the objectives space (i.e., point $\boldsymbol{x_1^*}$ in the decision space). Then, the second objective is optimized with the additional constraint $\beta \leq 37.57$ by which we obtain the optimal solution (37.57, 148) in the objectives space (i.e., the first row of payoff table) which is a non-dominated solution dominating the previous one.

For constructing the second row of payoff table, we first optimize the second objective over the feasible region by which we obtain the optimal solution (40.94, 180) in the objectives space. Then, the first objective is optimized with the additional constraint $\sum_{h=1}^{H}\sum_{q=1}^{Q}\sum_{t=1}^{T}\lambda_{tq}.z_{tqh} \geq 180$ by which we obtain the same optimal solution which ensures that it is a non-dominated solution.

Consequently, the payoff table is constructed as follows:

Table 5 Payoff table

	Z_1	Z_2
$\boldsymbol{x_1^*}$	37.57	148.00
$\boldsymbol{x_2^*}$	40.94	180.00

After the construction of payoff table, we divide the range of the second objective function to four equal intervals and we use the resulting five grid points as the values of e_2. Hence, vector $\boldsymbol{e_2}$ is written as $\boldsymbol{e_2}$ = (148, 156, 164, 172, 180). Now for each components of $\boldsymbol{e_2}$ the following model is solved:

$$min\left(\beta - eps\left(\frac{s_2}{r_2}\right)\right)$$

s.t.

$$\sum_{h=1}^{H}\sum_{q=1}^{Q}\sum_{t=1}^{T}\lambda_{tq}.z_{tqh} - s_2 = e_2 \qquad (23)$$

Constraint (16)

Constraints (3)-(13)

$s_2 \in R^+$

Where r_2 denotes the range of second objective function from the payoff table which is equal to 32. In this manner, for each given value of e_2, the optimal solution of above model will certainly generate an

efficient solution. Table 6 shows the resulting non-dominated solutions.

Table 6 Non-dominated solutions found by AUGMECON

Z1 (min)	Z2 (max)
37.57	148.00
37.74	159.00
38.27	168.00
39.62	174.00
40.94	180.00

It is noteworthy that the early exit option does not require for our problem because our problem is a bi-objective one and we vary only one RHS value, therefore, we do not have nested loops in this case. Furthermore, the augmented epsilon-constraint version of the proposed model was coded in GAMS and the CPLEX 7.5 solver was used for solving the corresponding single-objective models on a 2.0 GHz Dual Core CPU with 1GB of RAM. The above sample problem was solved within the 4.12 seconds of CPU time.

6 Concluding remarks

In this paper, a novel bi-objective mathematical model is proposed to make the best decisions relating to the heads of multi-head beam-type placement machines. Due to the importance of the heads from the machine performance point of view, optimizing their performance has a great effect on the whole production process. In the present study, a criterion, namely appropriateness function, is presented for the first time to evaluate the compatibility of each pair of component type-nozzle. The selection of an appropriate set of nozzles for handling the components on the heads enables the robotic arm to move faster. Hence, the machine cycle time can considerably be improved by adopting appropriate nozzles to the heads. In order to offer several efficient solutions to the decision maker before making his/her final decision, we apply an improved version of a well-known multi-objective solution method, i.e., the epsilon constraint method called augmented epsilon constraint method (AUGMECON). Although the efficient solutions of the proposed model could be found using the AUGMECON method by applying the commercial optimization solvers like CPLEX, it should be noted that the corresponding computational time grows exponentially with the problem size. Therefore, in order to solve the real-sized problem instances more efficiently, developing appropriate heuristic or meta-heuristic solution methods is of great interest. Considering the synchronous nozzle exchanges on the heads can also be used to define the new problem scenarios.

Acknowledgement

This study was supported by the University of Tehran under the research grant No. 8109920/1/03. The authors are grateful for this financial support.

References

1. Hardas C, Doolen T, Jensen D (2008) Development of a genetic algorithm for component placement sequence optimization in printed circuit board assembly. Comput ind eng 55:165-182
2. Ayob M, Kendall G (2008) A survey of surface mount device placement machine optimization: Machine classification. Eur J Oper Res 186:893-914
3. Sun D-S, Lee T-E (2008) A branch-and-price algorithm for placement routing for a multi-head beam-type component placement tool. OR Spectr 30:515-534

4. Crama Y, Klundert J, Spieksma F-C-R (2002) Production planning problems in printed circuit board assembly. Discrete Appl Math 123:339-361
5. Lee W, Lee S, Lee B, Lee Y (2000) An efficient planning algorithm for multi-head surface mounting machines using a genetic algorithm. J Univers Comput Sci 5:833-854
6. Ball M-O, Magazine M-J (1988) Sequence of insertions in printed circuit board assembly. Oper Res 36:192-201
7. Leipala T, Nevalainen O (1989) Optimization of the movements of a component placement machine. Eur J Oper Res 38:167-177
8. Or I, Duman E (1996) Optimization issues in automated production of printed circuit boards operations sequencing, feeder configuration and load balancing problems. Proc IEEE Emerging Technologies and Factory Automation 227-232
9. Khoo L, Loh, K (2000) A genetic algorithms enhanced planning systems for surface mount PCB assembly. Int J Adv Manuf Technol 16:289-296
10. Ho W, Ji P (2005) A genetic algorithm to optimize the component placement process in PCB assembly. Int J Adv Manuf Technol 26:1397-1401
11. Moon G (2009) Efficient operation methods for a component placement machine using the patterns on printed circuit boards. Int J Prod Res DOI: 10.1080/00207540802553608
12. Van Laarhoven P-J-M, Zijim W-H-M (1993) Production preparation and numerical control in PCB assembly. Int J Flex Manuf Syst 5:187-207
13. Magyar G, Johnson M, Nevalainen O (1999) On solving single machine optimization problems in electronics assembly. J Electron Manuf 9:249-267
14. Hong J, Lee W, Lee S, Lee B, Lee Y (2000) An efficient production planning algorithm for multi-head surface mounting machines using the biological immune algorithm. Int J Fuzzy Syste 2:45-53
15. Jeevan K, Parthiban A, Seetharamu K-N, Azid, Quadir G-A (2004) Optimization of PCB component placement using genetic algorithms. J Electron Manuf 11:69-79
16. Grunow M, Gunther H-O, Schleusener M, Yilmaz I-O (2004) Operations planning for collect-and-place machines in PCB assembly. Comput ind eng 47:409-429
17. Sun D-S, Lee T-E, Kim K-H (2005) Component allocation and feeder arrangement for a dual-gantry multi-head surface mounting placement tool. Int J Prod Econ 95:245-264
18. Raduly-Baka C, Knuutila (2007) Selecting the nozzle assortment for a Gantry-type placement machine. OR Spectr 30:493-513
19. Kulak O, Yilmaz I-O, Gunther H-O (2007) PCB assembly scheduling for collect-and-place machines using genetic algorithms. Int J Prod Res 45:3949-3969
20. Li S, Hu C, Tian F (2008) Enhancing optimal feeder assignment of the multi-head surface mounting machine using genetic algorithms. Appl Soft Comput 8:522-529
21. Ayob M, Kendall G, (2009) The optimisation of the single surface mount device placement machine in printed circuit board assembly: a survey, Int J Systems Science, 40:553–569.
22. Miettinen K (1999) Nonlinear multi-objective optimization. Kluwer Academic Publishers, Boston
23. Mavrotas G (2009) Effective implementation of the ε-constraint method in Multi-Objective Mathematical Programming problems. Appl Math Comput, 213:455-465
24. Ehrgott M, Wiecek M (2005) Multiobjective Programming in. In: J Figueira, S.Greco, M

Ehrgott (eds) Multiple Criteria Decision Analysis
State of the Art Surveys. Springer, 667-722