A Cost-oriented Model for Multi-manned Assembly Line Balancing Problem

Abolfazl Kazemi[a,*], Abdolhossein Sedighi[b]

[a] *Assistant Professor, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran*

[b] *Msc, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran*

Received 5 August, 2012; Revised 20 January, 2013; Accepted 13 March, 2013

Abstract

Many real-world production systems produce large-sized commodities. Due to the size of the production unit, it is typical to see that more than one worker is working on the same work-piece. This type of assembly line in which multiple workers operate on the same work-piece simultaneously is called multi-manned assembly line (MAL). In the classical multi-manned assembly line balancing problem (MALBP) the objective is to minimize the manpower needed to manufacture one product unit. Apart from the manpower, other cost drivers like wage rates or machinery are neglected in this classical view of the problem.

However due to the high competition in the current production environment, reducing the production costs and increasing utilization of available resources are very important issues for manufacturing managers. In this paper a cost-oriented approach is used to model the MALBP with the aim of minimizing total cost per production unit. A mathematical model is developed to solve the problem. Since the proposed model is NP-hard, several heuristic algorithms and a genetic algorithm (GA) are presented to efficiently solve the problem. Parameters and operators of the GA are selected using the design of experiments (DOE) method. Several examples are solved to illustrate the proposed model and the algorithms.

Keywords: Multi-manned assembly line; Cost-oriented approach; Heuristic; Genetic algorithm; Design of experiments.

1. Introduction

Assembly lines are a type of production system used for mass production of standardized commodities. However, they can also be used to produce customized units in low volumes. In an assembly line several workstations (also called stations) are arranged along a conveyor belt. In each station a set of tasks are performed on the work-piece. Beginning from the first station, each work-piece is moved from station to station with a constant transportation speed throughout the line. The cycle time determines the production rate; specifically production rate is equal to the reciprocal of cycle time. The work content of each station in the line is constrained to be less than or equal to the cycle time. The total work needed to assemble the final product is divided into n basic operations I= {1, 2… n}, these elementary operations are called tasks. Each task j needs d_j^t units of time to be accomplished; this duration is called task time. Furthermore there are some precedence relations between tasks. Typically these relations are presented in a precedence graph in which each vertex presents a task and each arc (i,j) presents a precedence relation between tasks i and j.

The problem of partitioning tasks to stations in order to optimize some objective functions is called assembly line balancing (ALB) problem. The most studied problem in the field of ALB is the simple assembly line balancing problem (SALBP) and has the following assumptions (Baybars 1986; Scholl 1999; Scholl and Becker 2006):

- Mass production of one homogenous commodity.
- Given production process.
- Paced line with a fixed cycle time *C*.
- Each task j has a Deterministic and integer operation time d_j^t.
- No assignment constraints besides precedence constraints.
- Serial line layout with m stations.
- All stations are equally equipped with respect to machines and workers.
- Maximize the line efficiency: $Eff = \frac{t_{sum}}{m \times C}$ in which m is the number of stations and $t_{sum} = \sum_{j=1}^{n} d_j^t$ is the sum of processing time of all tasks.

These assumptions are very restricting with respect to real world assembly line systems. Therefore many researchers have focused on changing or releasing some of these assumptions to obtain more realistic models. The

*Corresponding author E-mail: abkaazemi@gmail.com

resultant problems are called generalized assembly line problem (GALBPs) (Boysen et al. 2007).

Numerous generalizations have been considered for the ALBP. Some examples of these generalizations are considering setup times between tasks(Andres et al. 2008; Martino and Pastor 2010; Seyed-Alagheband et al. 2011), considering resource constraints (Agpak 2005; Corominas et al. 2010), considering other objective functions that are more interesting in real world assembly lines (Pastor 2011), rebalancing assembly lines (Grangeon et al. 2011), considering U-shaped assembly lines balancing (Miltenburg and Wijngaard 1994), parallel workstations (Buxey 1974), considering process alternatives (Pinto et al. 1983) and two sided assembly lines (Bartholdi 1993). Some latest surveys of generalized assembly line problems are Erel and Sarin (1998), Rekiek et al. (2002), Scholl (1999), Scholl and Becker (2006).

In manufacturing large commodities it is beneficial to have several workers operate on the same work-piece in the same station. This situation is first identified and modeled by Dimitriadis (2006). In a MAL more than one operator can be working on the same work-piece in each station. This results in several advantages over simple assembly lines. Some examples of these advantages are reducing the length of the line and consequently reducing the work in process, reducing the costs of tools, machinery and also transportation system (Fattahi et al. 2011).

MALs are very popular production systems which are used in many industries that have a large sized work-piece. For instance this type of assembly system in which multiple workers are simultaneously operating on the same work-piece is very common in the final assembly of automobiles. In this environment bare painted car bodies arrive at the assembly line operation and complete cars are leaving it. Installation of different parts of the final car such as bumpers, seats, windows, wheels and so on, is performed by multiple workers on the same car body; this system is a MAL. In more automated environments multiple robotic arms are used. Robotic arms are an essential part of car manufacturing. Most industrial robots work in auto assembly lines, putting cars together. In these environments multiple robotic arms work simultaneously on the same work-piece and therefore the system can be considered a MAL. Other examples of this type of assembly system are cargo ship and airplane final assembly

Even though *MALs* are very common in real world assembly line production systems, only a small number of research papers have considered *MALBPs*. Dimitriadis (2006) introduced the *MALBP* and presented a heuristic assembly line balancing procedure to solve the problem. Cevikcan et al. (2009) developed a mathematical programming model to create assembly physical multi-manned stations in mixed model assembly lines. They also presented a scheduling-based heuristic to solve the problem. Chang and Chang (2010) proposed a mixed-model assembly line balancing problem with multi-manned workstations and developed a mathematical model for the mixed-model assembly line balancing problem with simultaneous production (*MALBPS*) to obtain the optimal number of workstations. They presented a coding system, Four-Position Code (*FPC*), to re-code the tasks in order to tackle with this issue, and they also provided a computerized coding program written in C++ to generate those *FPCs*. Fattahi et al. (2011) developed a mathematical programming model for *MALBP*. They also proposed an ant colony meta-heuristic approach to solve the problem.

In the literature of MAL usually the objective is to minimize the number of workers for a given cycle time or minimizing the idle time (Cevikcan et al. 2009; Chang and Chang 2010; Fattahi et al. 2011). However due to the high competition in the current production environment, reducing the production costs and increasing utilization of available resources are very important issues for manufacturing managers. Therefore developing a model to directly minimize the production costs is of significant interest. In this paper the MAL configuration is considered with a cost-oriented approach. Generally final assembly is a labour-intensive production (Amen 2006). In the cost-oriented approach the objective is to minimize the total cost per product unit (Amen 1997; Amen 2000a; Rosenberg and Ziegler 1992; Steffen 1997). Therefore the significant cost drivers should be analyzed.

At first the labour costs are considered. The payment of a worker is dependent on the "job values" determined by the well-known work measurement systems (Amen 2000a). In an assembly line there are tasks with different levels of difficulty and job values assigned to a worker. For each task i it is possible to consider a wage rate $k_i^{tw}, i \in I\ [^{MU}/_{TU}]$ (*TU* time unit, *PU* production unit, *MU* monetary unit) which is related directly to its job value. The wage rate of an operator working in a station along with other operators on the same work-piece, is determined by the most difficult task assigned to him (or her) i.e. the task with the highest job value. Therefore, the wage rate of worker l in station j is: $k_{jl}^{sw} = \max\{k_i^{tw} | i \in I_{jl}^s\}, j = 1,2 \dots J, l = 1,2 \dots MC\ [^{MU}/_{TU}]$ where I_{jl}^s is the set of tasks assigned to worker l in station j and *MC* is the maximum feasible concentration of workers in each station. It is important to note that the wage rates are paid for the total cycle time and not only for the sum of duration of tasks performed by the worker.

Furthermore, costs of capital should be considered. Examples of this kind of costs are machinery and material handling system e.g. conveyor. It is assumed that the costs of capital are directly dependent on the length of the line i.e. number of stations. The machinery needed to perform the operations can be a special machinery to perform a special task or a universal machinery. The number of special machinery can be assumed to be fixed and independent of assignment of tasks to the workers in the stations. In addition, it is assumed that all of the stations

need identical universal machines. Therefore the costs of capital for all stations are the same.

Other costs such as costs of material are assumed to be independent of the length of the line or assignment of tasks to stations (Steffen 1977). Therefore the total costs per product unit k [*MU*/*PU*] can be formulated as $k = \sum_{l\in L}\sum_{j\in J} C \times k_{jl}^{sw} + m \times k^{sc}$ where k^{sc} [*MU*/*PU*] is the total cost of capital.

Reviewing the literature of cost-oriented assembly line balancing, Rosenberg and Ziegler (1992) assumed that the operation of a station k causes a wage rate w_k per time unit equal to the maximum wage rate of all tasks that are assigned to that station. The objective is to minimize the aggregate wage rate over all stations, while the number of stations is variable. They described and evaluated priority rule-based heuristics, where some of the rules are available for SALBP-1. Amen (2000a) extended the problem by considering the costs of capital e.g. cost of machinery or transportation system. Amen (1997) proposed a branch and bound algorithm to solve the problem which applies a station-oriented construction method and laser search strategy. Amen (2000b) and Amen (2001) developed station-oriented priority rule based procedures with cost-oriented dynamic priority rules and compared them to existing ones using a large set of problem instances which is generated randomly. The new rule named "best change of idle cost" had a better performance than all other rules. For the same problem, Amen (2006) concentrated on general model formulations that can be solved by standard optimization tools and introduced several improvements to existent models. These models are designed for either general branch-and-bound techniques with LP-relaxation or general implicit enumeration techniques. They also discussed the solution difficulty of the problem and showed that the "maximally-loaded-station-rule" has to be replaced by the "two-stations-rule"; which causes an enormous increase in solution difficulty compared to the time-oriented version. Malakooti and Kumar (1996), Malakooti (1991) and Malakooti (1994) considered a multi-objective ALBP with objectives that are based on cost and capacity.

In this paper a cost-oriented approach is used to model the MAL which to the best of our knowledge hasn't been considered in the literature so far. Then different heuristics and a GA are proposed to solve the problem instances of large and medium size. The parameters and operators of the GA are selected using DOE method. The rest of this paper is organized as follows: in Section 1 the proposed model is described and a mathematical formulation is developed to solve the problem. Seven heuristic algorithms and a GA are proposed to solve the problem in Sections 2 and 3 respectively. Computational results are presented in Section 4. Finally the main conclusions of the paper and suggestions for future research are presented in Section 5.

2. Proposed Model and Mathematical Formulation

In this paper the paced assembly line with multi-manned workstations is considered which is very common in real world assembly lines but a small number of research papers have considered this type of assembly line. The work-piece stays at each station for a certain amount of time called cycle time. In each station there are several workers performing different tasks on the same work-piece. Each worker starts the tasks assigned to him (or her) as soon as it is technically possible. The main objective in this type of assembly line is to reduce the length of line while maintaining the effectiveness of the line. This type of multiple workers working on the same work-piece at the same time requires the work-piece to be of large size e.g. vehicle final assembly. Traditionally in simple assembly lines all of the tasks assigned to a worker can be performed continuously if the precedence relations are observed. But in multi-manned lines some tasks which are assigned to a worker may be delayed by the tasks assigned to other workers in the same workstation which is called unavoidable delay.

The objective is to minimize the total cost per production unit and the decisions involved in cost-oriented MALBP include the followings: (1) first how many workers should be assigned to each station then (2) which tasks to be performed by which worker. The notations used in the mathematical model is presented in

Table 1
Notations used in the mathematical model

i, h	task
j	station
l	worker
I	Set of tasks
L	Set of workers
J	Set of workstations
$P_i\ (P_i^*)$	Set of direct (all) predecessors of task *i*
$F_i\ (F_i^*)$	Set of direct (all) successors of task *i*
C	Cycle time ($TU\ /\ PU$)
m	Number of stations
M	A big positive number
MC	Maximum concentration of workers in a station
N	Number of tasks
d_i^t	Duration of task *i* when there are *k* workers in the station *(TU)*
k^{sc}	Cost of capital per station ($MU\ /\ PU$)
k_{jl}^{sw}	Wage rate of worker *l* in station *j*. ($MU\ /\ TU$)
k_i^{tw}	Wage rate of task *i*. ($MU\ /\ TU$)
$x_{ijl} \in \{0,1\}$	Equals 1 if task *i* is assigned to worker *l* in station *j*.
$y_{ih} \in \{0,1\}$	Equals to 1 if task i and h is assigned to the same worke and task *i* is performed earlier than task *h*.
st_i	Start time of task *i*

The problem under consideration is formulated as follows:

$$Min \left(\sum_{j\in J}\sum_{l\in L} J\times x_{Njl} \right) \times k^{sc} + \left(\sum_{l\in L}\sum_{j\in J} k_{jl}^{sw} \right) \times C \quad (1)$$

$$\sum_{j\in J}\sum_{l\in L} x_{ijl} = 1 \qquad \forall i \in I \quad (2)$$

$$\sum_{j\in J}\sum_{l\in L} J\times x_{hjl} \le \sum_{j\in J}\sum_{l\in L} J\times x_{ijl} \qquad \forall i \in I,\ h \in P_i \quad (3)$$

$$st_i + d_i^t \le C \qquad \forall i \in I, j \in J \quad (4)$$

$$st_i - st_h + M \times \left(1 - \sum_{l\in L} x_{hjl}\right) + M \times \left(1 - \sum_{l\in L} x_{hjl}\right) \ge d_h^t \qquad \forall i \in I, h \in P_i, j \in J \quad (5)$$

$$st_h - st_i + M \times (1 - x_{hjl}) + M \times (1 - x_{ijl}) + M \times (1 - y_{ih}) \ge d_i^t \qquad \forall i \in I, j \in J, l \in L,\ h \in \{r \mid r \in I\text{-}(P_i^* \cup F_i^*) \wedge i<r\} \quad (6)$$

$$st_i - st_h + M \times (1 - x_{hjl}) + M \times (1 - x_{ijl}) + M \times (y_{ih}) \ge d_h^t \qquad \forall i \in I, j \in J, l \in L,\ h \in \{r \mid r \in I\text{-}(P_i^* \cup F_i^*) \wedge i<r\} \quad (7)$$

$$\sum_{i\in I} x_{ij,l+1} \le N \times \sum_{i\in I} x_{ijl} \qquad \forall j \in J, l \in L \quad (8)$$

$$k_i^{tw} \times x_{ijl} \le k_{jl}^{sw} \qquad \forall i \in I, j \in J, l \in L \quad (9)$$

$$st_i \ge 0 \qquad \forall i \in I \quad (10)$$

$$k_{jl}^{sw} \ge 0 \qquad \forall j \in J, l \in L \quad (11)$$

$$x_{ijl} \in \{0,1\} \qquad \forall i \in I, j \in J, l \in L \quad (12)$$

$$y_{ih} \in \{0,1\} \qquad \forall i \in I,\ h \in \{r \mid r \in I - (P_i^* \cup F_i^*) \wedge i<r\} \quad (13)$$

In this formulation, equation (1) indicates the objective function should to be minimized which is the total cost per production unit. The first term presents the cost of capital which equals to the number of stations multiplied by k^{sc} (cost of capital per station). It is assumed that the task N is successor of all of the tasks in the precedence graph; if that's not the case a fictitious task with zero duration and wage rate must be considered. Therefore task N is always assigned to the last station. The second term in equation (1) presents the costs of labour which is the sum of wage rates of all workers in all stations multiplied by cycle time. Constraint (2) implies that each task i must be assigned to exactly one worker in one station. Constraint (3) ensures that precedence relations are observed. Equation (4) implies that all tasks must be finished before the cycle time. Equation (5) indicates that if task h is a direct predecessor of task i and they both assigned to the same station then starting time of task i must be greater than or equal to the finish time of task h. Constraint pair (6) and (7) is disjunctive for large enough values of M. This means that only one of them is active at the same time. Only when tasks i and h don't have any precedence relation and are both assigned to the same worker in the same station they become active. If yih=0 equation (6) becomes redundant and equation (7) implies that $st_i \ge st_h + d_h^t$ implying that task i must be scheduled after task h. On the other hand if yih=1 equation (7) becomes redundant and equation (6) implies that $st_h \ge st_i + d_i^t$ indicating that task h must be scheduled after task i. Constraint (8) indicates that in each station, workers are used in an increasing order of their indexes. Equation (9) implies that among all tasks assigned to worker l in station j the maximum wage rate is set to be the wage rate of the worker. Equations (10) and (11) ensure that start times and wage rates are non-negative. Equations (12) and (13) indicate that xijl and yih are binary variables.

3. Heuristic Algorithms Developed

Since the traditional cost oriented assembly line balancing problem is NP-hard (Amen 2000a; Rosenberg and Ziegler 1992) and the problem considered here is a generalization of it, the problem considered in this paper is also NP-hard. In other words by setting L=1 and $k^{sc} = 0$, the problem under consideration is reduced to the well-known cost oriented assembly line balancing problem which is NP-hard, therefore the considered

problem is also NP-hard. Therefore it is justified to develop heuristic algorithms to obtain good solutions in a computational time short enough to be applied in industrial real instances.

In this paper seven priority rule based heuristic algorithms are presented to solve the problem under consideration. These rules are as follows:

- Max_D: maximum task duration (Tonge 1965).
- Max_R: maximum ranked positional weight which is $max\{r_i | i \in I\}$ where $r_i = \begin{cases} d_i^t + \sum_{j \in F_i} r_j \,, if\ F_i \neq \emptyset \\ d_i^t \qquad\qquad if\ F_i \neq \emptyset \end{cases}$ (Hahn 1972; Helgeson and Birnie 1961).
- Max_F: maximum number of immediate followers in the precedence graph (Tonge 1965).
- Max_Kt: maximum cost rate (Rosenberg and Ziegler 1992).
- Min_Kt: minimum cost rate (Steffen 1977).
- Min_Kts: minimal absolute difference to the workers current cost rate i.e. $min\{|k_i^t - k_{jl}^{sw}| | i \in I\}$. This rule is a modification of the rule proposed by Steffen (1977).
- Min_Ki: best change of idle cost i.e. $min\{\Delta k_i | i \in I\}$; where $\Delta k_i = \begin{cases} -d_i^t k_i^t, & if\ k_i^t \leq k_{jl}^{sw} \\ (k_i^t - k_{jl}^{sw})C - d_i^t k_i^t, & if\ k_i^t > k_{jl}^{sw} \end{cases}$. This rule is a modification of the rule proposed by Amen (2000b).

The first five rules are static; this means that the priority of tasks doesn't change throughout building a solution. All of the static rules use a main procedure to assign tasks to workers in each station. This procedure is as follows:

Step 1: Set the current station *Sc*=1, and available tasks *Avail_task*= {1, 2… *N*}. Available tasks are the tasks that haven't been assigned to any worker in any station.

Step 2: Set the number of workers in the station *Wn*=1 and number of *Tc*=0.

Step 3: Among tasks of *Avail_task*, ones that are assignable to station *Sc*, select the task with the highest priority, according to one of the priority rules which will be presented later in this section. Assign it to the worker that starts the task earlier ties are broken in favor of the worker that is not idle i.e. assigning the task to the worker does not lead to unavoidable idle times. The final tie breaker is the index of the worker and the worker with lower index has more priority than the one with higher index. Delete the task from *Avail_task* then set *Tc*=*Tc*+1. If the selected worker is empty then set *Wn*=*Wn*+1. Repeat this step until there is no task assignable to station *Sc*. Then go to step 4. A task is assignable to a station if it has no predecessor in *Avail_task* and assigning it to the station doesn't violate the cycle time constraint.

Step 4: A station *Wn* number of workers is completed. If *Avail_task* is empty end the procedure, otherwise set *Sc*=*Sc*+1 and go to step 2.

The last two rules are dynamic and the priority of tasks may change throughout building a solution. These rules are also dependent on the worker to whom the task is assigned. Therefore another procedure is needed to build a solution with these rules. This procedure is as follows:

Step 1: Set the current station *Sc*=1, and available tasks *Avail_task*= {1, 2… *N*}. Available tasks are the tasks that haven't been assigned to any worker in any station.

Step 2: Set the number of workers in the station *Wn*=1 and number of *Tc*=0.

Step 3: Among tasks of *Avail_task*, ones that are assignable to station *Sc*, select the task with the highest priority. This involves selecting a pair (i, l) of task *i* and worker *l* which has the highest priority. Ties are broken in favor of the pairs that don't create idle times; second level ties are broken in favor of lower worker indexes. Finally the third level ties are broken in favor of lower task indexes. Delete the task from *Avail_task* then set *Tc*=*Tc*+1. If the selected worker is empty then set *Wn*=*Wn*+1. Repeat this step until there is no task assignable to station *Sc*. Then go to step 4. A task is assignable to a station if it has no predecessor in *Avail_task* and assigning it to the station doesn't violate the cycle time constraint.

Step 4: A station *Wn* number of workers is completed. If *Avail_task* is empty end the procedure, otherwise set *Sc*=*Sc*+1 and go to step 2.

Therefore five static and two dynamic rules are presented in this section to solve the problem. In the next section a genetic algorithm is proposed.

4. Proposed GA

Genetic algorithm is a strong meta-heuristic used in many combinatorial optimization problems. Many researches have successfully used genetic algorithm to solve generalized assembly line balancing problem among who we can mention (Akpinar and Bayhan 2011; Yolmeh and Kianfar 2012). In a GA every individual (or chromosome) is encoded by a structure that represents its properties, a population of individuals are produced and then each individual in the populations is evaluated and a fitness value is assigned to it. This value represents quality of the individual i.e. the higher fitness value, the better solution. Afterwards on the basis of a selection method, a set of individuals are selected as parents for

crossover and mutation to produce a new population. This process, which is called generation, is iterated until stopping criteria are met. The elements of the proposed GA are described in the following sections.

4.1. Solution encoding and initial population

In the proposed GA, an individual is represented by a permutation of tasks that shows the relative priority among tasks. In other words the leftmost task in the permutation has the highest priority and the rightmost task has the lowest priority. Fig. 1 shows an example of this representation. As it can be seen from this figure task 5 has the highest priority among all tasks, task 6 has the second high priority and task 7 has the least priority. It is important to note that since the chromosomes only determine the priority of tasks, observing the precedence constraints is not necessary. In other words the order in which the tasks appear in the chromosome is not necessarily the order in which they are assigned to the workers in the stations.

5	6	2	1	4	3	7

Fig. 1 an example of solution representation

To evaluate an individual, using the first procedure described in Section 2, a solution is created and the corresponding cost is computed. In simple Gas, initial population is generated randomly, but in improved Gas, the result of some heuristic or meta-heuristics is added to the initial population. Generally, adding the results of other heuristic or meta-heuristics, genetic algorithms could obtain better solutions in a shorter time and with better convergence speed. Two alternatives are considered for the initial population: (1) the whole population is generated randomly, (2) all of the priority rule based heuristics proposed in Section 2 is added to the initial population.

4.2. Selection mechanism and replacement

In this paper, the Roulette wheel method is used for selection procedure(Sivanandam and Deepa 2008). Applying crossover and mutation operators may result in losing the best known solution so far. To avoid this issue, a percentage of best solutions in the current population are directly transferred to the next population, this process is called replacement. By applying replacement procedure, the best solutions of the new population will not be worse than that of previous populations. In this paper, 10 percent of the best individuals in the current population are directly transferred to the next population.

4.3. Crossover

Crossover takes two parents and combines their characteristics to produce an offspring. Therefore the resulting child will inherit its characteristics from the parents. Crossover operator is applied to the mating pool with the hope of creating a better child. In the proposed GA, three types of crossover methods are considered. The first considered operator is called "OX or ordered crossover"; the second and third ones are proposed by Ruiz et al. (2005) and are called "similar job order crossover" or SJOX and "similar block order crossover" or SBOX, respectively. These crossover operators were used in the flow shop and single machine scheduling problems (Bahalke et al. 2010; Ruiz et al. 2005).

4.4. Mutation

After crossover process, the chromosomes are subjected to mutation. Mutation prevents the algorithm to be trapped in a local optimum. Mutation plays an important role in searching the whole search space by introducing unexplored areas. Mutation is considered as a background operator to maintain diversity in the population. In this paper three well-known types of mutation operators are used:

- Scramble mutation: two positions are selected randomly and the tasks between these positions are rebuilt randomly.
- Swap mutation: two tasks are selected randomly and swap their positions.
- Shift mutation: a randomly selected task is relocated to a randomly selected position and the tasks between these two positions move along.

1.1. Restart scheme

In GA after some iteration, the population diversity may be sufficiently low to converge to a local optimum. To avoid this drawback a process which is called restart scheme is used. This method has been used for flow shop and single machine scheduling problems(Bahalke et al. 2010; Ruiz et al. 2005; Ruiz et al. 2006; Vallada and Ruiz 2010) . In this process if the best known solution so far has not improved for G_r consecutive generations, the whole population is reproduced. The restart scheme applied in this paper works as follows:

1) Set $Count = 0$.
2) In each iteration I store the minimum cost: $Cost_i$
3) If $Cost_i = Cost_{i-1}$ then $Count = Count + 1$ else set count=0.
4) If $Count > G_r$ do the followings:
   - Sort the population in ascending order of cost.
   - Skip the first 20% of the population.
   - 50% of the population is produced by doing scramble mutation on the first 20% best chromosomes.

- The remaining 30% of the population is generated randomly.

4.5. Selecting the parameters and operators of GA

In this section the parameters and operators of the GA are selected using the design of experiments (DOE) method. DOE is a structured method that determines the relationship between the factors that affect output of a process. In order to calibrate the algorithm, a full factorial design is chosen i.e. all possible combinations of the following factors are tested:

- Initial population (Init_pop): 2 levels (using the results of other heuristic or not).
- Population size (Pool_size): 3 levels (20, 40 And 60).
- Crossover operator (Cross_type): 3 levels (OX, SJOX and SBOX).
- Mutation operator (Mut_type): 3 levels (scramble, swap and shift).
- Mutation probability (Mut_prob): 3 levels (0.05, 0.1 and 0.15).
- Gr in the restart scheme (Gr): 3 levels (5, 10 and M).

All of the above mentioned factors result in $2 \times 3 \times 3 \times 3 \times 3 \times 3 = 486$ different combinations and thus 486 different genetic algorithms. The following procedure is applied to produce problem instances. First, the precedence network (i.e., precedence graph) with the desired network density (i.e. order strength) is produced using a problem generator (Gehrlein 1986). Only the precedence graphs with network densities that fall within the target range of ± 0.05 are selected and processing times and wage rates of the workers are randomly generated for each task using a discrete uniform distribution with minimum and maximum values of 5 and 30, respectively. Cycle time is randomly selected from one of the values 30, 45 and 60. Finally the maximum concentration of workers is randomly selected between values 3, 4 and 5. To produce a problem set following three levels of N (number of tasks) and OS (order strength) are considered.

- N: 10, 30 and 50.
- OS: 0.25, 0.5 and 0.75.

For each level of N and OS two instances are generated. Therefore a total of 18 $(3 \times 3 \times 2)$ instances are generated. The stopping criterion for the GA is set to a CPU time limit of $200 \times N$ milliseconds. The algorithm is coded in C language and runs on a personal computer with Intel Pentium dual processor 2.2 GHz and 2 GB of RAM. The response variable for this experiment is calculated as follows:

$$Relative\ deviation\ (RD) = \frac{Algorithm_{sol} - LB}{LB} \quad (14)$$

Where $Algorithm_{sol}$ is the solution obtained by a given algorithm on a given instance, LB is the lower bound for the instance. To calculate a lower bound on the total costs, lower bounds on the costs of capital and the costs of labor are needed. At first the lower bound for the costs of capital is explained. To calculate a lower bound on the costs of capital a lower bound on the number of stations is needed. It is assumed that the first task in the precedence graph is predecessor of all other tasks. Similarly it is assumed that the last task in the graph is successor of all of the other tasks. If there is no such tasks, fictitious tasks is to be considered. To obtain a lower bound on the number of stations, the longest path, also called critical path, from the first task to the last task is considered. The length of this path is a lower bound on the time needed to produce one commodity, lessening or increasing the number of workers in each station does not change this value. Thus, the formulation for lower bound is:

$$LB_{station} = \left\lceil \frac{\sum_{j \in critical\ path} t_j}{C} \right\rceil \quad (15)$$

Therefore a lower bound on the costs of capital is obtained using the following formulation:

$$LB_{capital\ cost} = LB_{station} \times k^{sc} \quad (16)$$

To obtain a lower bound on the costs of labor at first a lower bound on the number of workers is calculated using the formulation: [16].

$$LB_{worker} = \left\lceil \frac{\sum_{j \in I} t_j}{C} \right\rceil \quad (17)$$

Therefore at least LB_{worker} workers are needed. The lower bound on the costs of labor can be computed using the following formulation:

$$LB_{labor\ cost} = C \times MWR \quad (18)$$

In this formulation MWR is the sum of LB_{worker} smallest wage rate values. Therefore the lower bound on the costs of production is computed using the following formula:

$$LB = LB_{capital\ cost} + LB_{labor\ cost} \quad (19)$$

The experiment was analyzed by means of a multifactor ANOVA technique. For results of ANOVA to be valid three hypotheses should be checked: normality, homoskedasticity and independence of residuals. The resultant residuals from the experimental data were analyzed and all three hypotheses were valid. Table 1 shows the results of ANOVA for this experiment, all interactions of more than two factors have been ignored as their F value was too small.

In ANOVA if the p-value is less than the considered significance level α, different levels of the corresponding factor have a significant effect on the response variable.

In this study the significance level α is assumed to be 0.05. Since p-values in the table are very small and close to zero, the analyses are made on the basis of F-ratio. F-ratio is the ratio between the variance explained by a factor and unexplained variance, the greater this value the more effect of the factor on the response variable. The procedure of selecting operators and parameters for GA is as follows: starting from the greatest F-ratio for a factor or interaction, the preferred level is selected for the corresponding factor. Then the next factor with the maximum value of F-ratio is considered and the same process is applied, this procedure is repeated until suitable levels for all factors are selected.

As it can be seen in Table 2, the two problem dependent factors, N and OS, have the greatest values of F-ratio. This is expected, because as the size and order strength increases, difficulty of the instance increases and this leads to more deviation from the lower bound. The next value of F-ratio corresponds to crossover type (Cross_type), as it can be seen from the main effects plot in Fig. 2 the desired level for this factor is 3, which corresponds to the SBOX crossover. The third greatest value of F-ratio corresponds to population size and as it can be seen from the main effects plot in Fig. 2 the desired level for population size is 3 which corresponds to a population size of 60, applying this procedure to all factors, using the main effects plot presented in Fig. 2, and the interaction plot in Fig. 3, the following levels are obtained:

- Initial population (Init_pop): using the results of other heuristic.
- Population size (Pool_size): 60.
- Crossover operator (Cross_type): SBOX.
- Mutation operator (Mut_type): shift mutation.
- Mutation probability (Mut_prob): 0.15.
- Gr in the restart scheme (Gr): 10.

Table 1
Results of ANOVA for the GA

Source	DF	Seq SS	Adj SS	Adj MS	F	P
N	2	44.2303	44.2303	22.1152	8104.37	0
OS	2	7.073	7.073	3.5365	1295.99	0
Cross_type	2	1.5717	1.5717	0.7859	287.98	0
Mut_type	2	0.007	0.007	0.0035	1.29	0.275
Pool_size	2	0.3274	0.3274	0.1637	60	0
Mut_prob	2	0.0122	0.0122	0.0061	2.23	0.108
Init_pop	1	0.0467	0.0467	0.0467	17.12	0
Gr	2	0.0501	0.0501	0.0251	9.18	0
Cross_type*Mut_type	4	0.0621	0.0621	0.0155	5.69	0
Cross_type*Pool_size	4	0.0344	0.0344	0.0086	3.15	0.014
Cross_type*Mut_prob	4	0.0032	0.0032	0.0008	0.29	0.883
Cross_type*Init_pop	2	0.0028	0.0028	0.0014	0.52	0.595
Cross_type*Gr	4	0.0016	0.0016	0.0004	0.15	0.965
Mut_type*Pool_size	4	0.0526	0.0526	0.0132	4.82	0.001
Mut_type*Mut_prob	4	0.0023	0.0023	0.0006	0.21	0.933
Mut_type*Init_pop	2	0.0007	0.0007	0.0004	0.13	0.878
Mut_type*Gr	4	0.002	0.002	0.0005	0.18	0.949
Pool_size*Mut_prob	4	0.002	0.002	0.0005	0.18	0.947
Pool_size*Init_pop	2	0.0013	0.0013	0.0007	0.24	0.784
Pool_size*Gr	4	0.0073	0.0073	0.0018	0.67	0.615
Mut_prob*Init_pop	2	0.0011	0.0011	0.0006	0.21	0.814
Mut_prob*Gr	4	0.0011	0.0011	0.0003	0.1	0.981
Init_pop*Gr	2	0.0033	0.0033	0.0016	0.6	0.551
S=0.052238	R-Sq=97.35%		R-Sq(Adj)=97.28%			

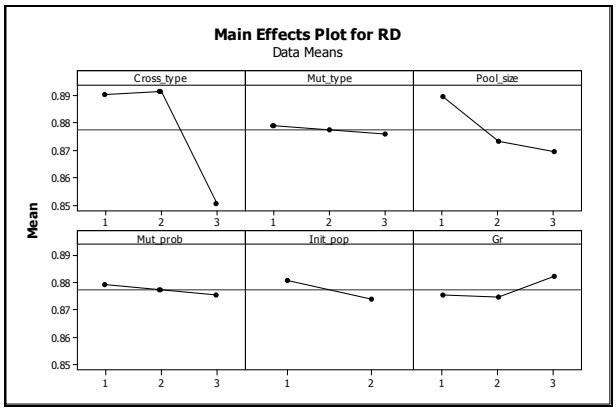

Fig. 2. Main effects plot for the GA

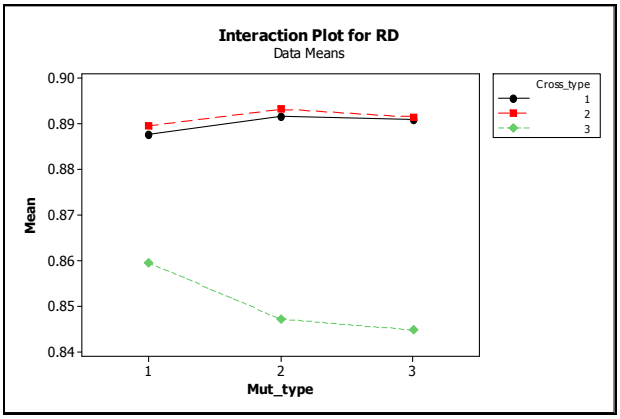

Fig. 3. Interaction plot between Cross_type and Mut_type factors

5. Computational Results

In this section computational experiments are presented. All of the algorithms are coded in C language and runs on a personal computer with Intel Pentium dual processor 2.2 GHz and 2 GB of RAM. Parameters and operators of the GA are selected according to the results obtained in Section 3.6. The stopping criterion when testing algorithms with instance is set to a CPU time limit of $200 \times N$ milliseconds.

In the first experiment the motivation is comparing the cost-oriented model with the traditional time-oriented model. Therefore an example is presented and solved with both time-oriented and cost-oriented approaches. The precedence graph and task times for this example are taken from the well-known instance of Mertens which is available at www.assembly-line-balancing.de. In the time oriented model at first the number of workers is minimized as the primary objective and then the number of stations is minimized as the secondary objective. The precedence graph of this example with duration and wage rate of tasks is shown in Fig. 4. The cycle time and maximum feasible worker concentration in each station for this instance is assumed to be 8 and 3 respectively. Also total cost of capital per station (k^{sc}) is assumed to be equal to 5.

Optimum solutions for cost-oriented and time-oriented versions of this problem are presented in Figures 5 and 6 respectively. In these figures for each task, starting time and finishing time are shown alongside its bar. Shaded rectangles designate idle time at the end of the cycle time.

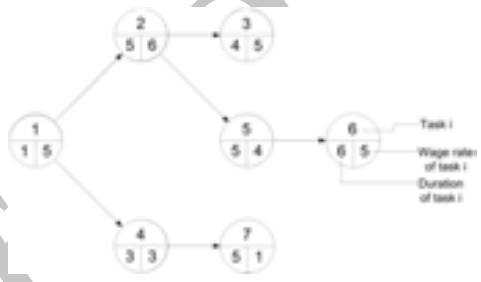

Fig. 4. example of a problem instance

Table 3 shows the cost calculations for the optimal solutions obtained by time-oriented and cost-oriented approaches. As seen from this table a total of 199 monetary units are needed to produce one production unit are being used in the traditional line balancing model, while this number could be reduced to 183 with the proposed model. Thus, 16 monetary units are saved. Besides, the required number of workers and stations are calculated as 5 and 3 respectively. These numbers are the same with the ones obtained by the time-oriented model, which means that the solution is also optimal in terms of number of workers and stations. Consequently, the solution is the best in terms of the total cost, and while reaching this best, the best number of stations and workers are also achieved.

In the second experiment the performance of the proposed algorithms is illustrated. To do so, each of the 25 different precedence graphs available at www.assembly-line-balancing.de is used to generate an instance. For each task in each instance the wage rate of task i is assumed to be: $k_i^{tw} = d_{N+1-i}^{t}$. The cycle time is generated randomly between maximum task time tmax and 2* tmax. The cost of capital for each station is assumed to be: $k^{sc} = \frac{c^2}{2}$. Each instance is solved by the proposed algorithms (the GA is run once for each instance) and the relative deviation is computed using equation (1) in Section 3.6. The results are presented in Fig. 7.

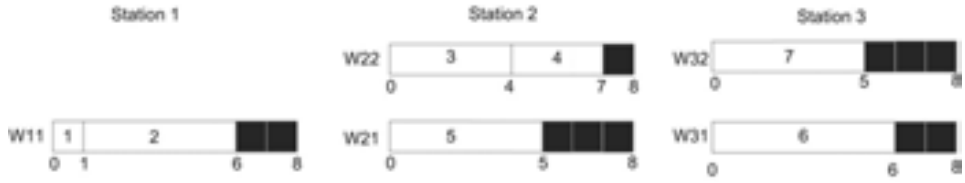

Fig. 5. The optimum solution for cost-oriented approach

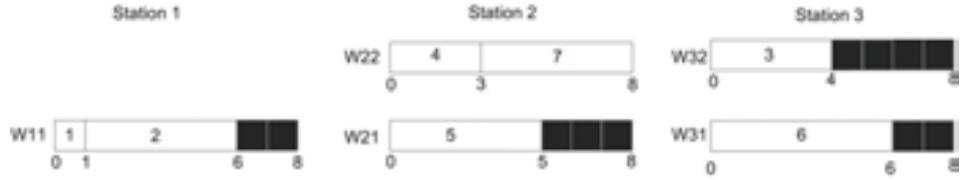

Fig. 6. The optimum solution for time-oriented approach

Table 2
Optimal solutions to the example

Station	Worker	Time-oriented optimal solution		Cost-oriented optimal solution	
j	l	I_{jl}^{s}	k_{jl}^{sw}	I_{jl}^{s}	k_{jl}^{sw}
1	1	{1,2}	6	{1,2}	6
2	1	{5}	4	{5}	4
	2	{4,7}	3	{3,4}	5
3	1	{6}	5	{6}	5
	2	{3}	5	{7}	1
$\sum_{l\in L}\sum_{j\in J} k_{jl}^{sw}$			23		21
m			3		3
$k=\sum_{l\in L}\sum_{j\in J} C\times k_{jl}^{sw}+m\times k^{sc}$			199		183

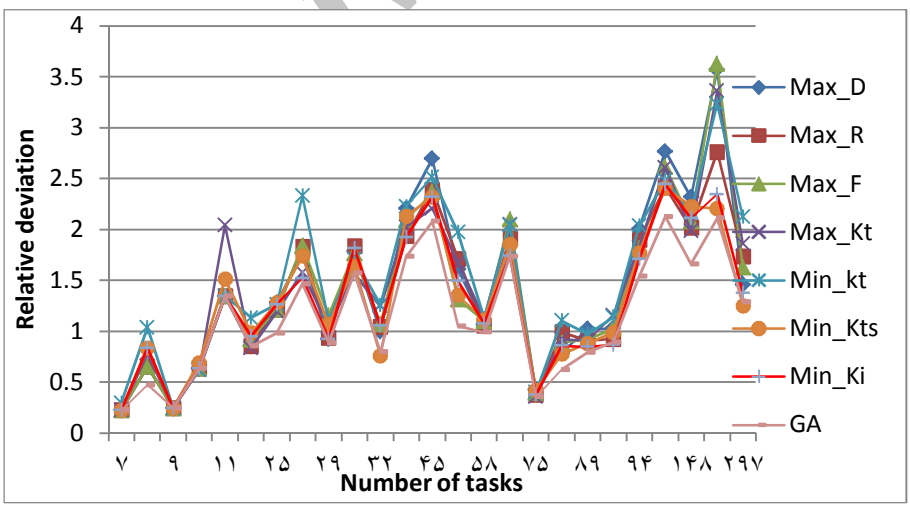

Fig. 7 performance of the proposed heuristic methods

As seen in Fig. 7 the two dynamic priority rules, Min_Kts and Min_Ki, have a better overall performance comparing to other priority rules. This highlights the importance of considering the current cost rate of the workers while building the solution. Furthermore, as expected, the GA performs significantly better than other algorithms. This provides the motivation to perform a final experiment to illustrate the performance of the proposed GA and provide a background to compare the results of the GA with other future algorithms. For this experiment another data set is generated using a selection of well-known instances for *SALBP*-1. In order to facilitate comparison of the proposed algorithm with other future algorithms, the wage rates for each task: $k_i^{tw} =$

d^t_{N+1-i} and cost of capital for each station is set to be: $k^{sc} = \frac{c^2}{2}$. For each instance the GA is run for 5 iterations and the average cost and the computed lower bound for the instance are reported in Table 4. The optimum numbers of stations for Mertens, Bowman and JAESCHKE examples are obtained by solving the mathematical model using Lingo 11. For these instances the optimum solutions found by the GA are marked with a star. For other examples a lower bound is computed using equation (6) in Section 5.4.

Table 3
Results of *the GA* for a selection of well-known instances

Author	Tasks	Cycle time	Optimal number of stations for *SALBP*	Optimal number of stations and workers (or a lower bound of it)	Average Obtained number of stations and workers	Average Space utilization (%)	Optimal cost (or a lower bound of it)	Average Obtained cost	MC
MERTENS	7	6	6	3,6	3,6	50.0	198	198.0*	4
		7	5	3,5	3,5	60.0	220.5	220.5*	4
		8	5	3,5	3,5	60.0	264	264.0*	4
		10	3	3,3	3,3	100.0	300	300.0*	4
		15	2	2,2	2,2	100.0	390	390.0*	3
BOWMAN8	8	20	5	4,5	4,5	80.00	1820	1820*	4
JAESCHKE	9	6	8	5,7	6,8	75.0	306	306.0*	4
		7	7	6,7	6,7	85.7	371	371.0*	4
		8	6	5,6	5,6	83.3	368	368.0*	4
		10	4	3,5	3,5	60.0	360	360.0*	4
		18	3	2,3	2,3	66.7	540	540.0*	4
JACKSON	11	7	8	4,7	5,9	55.6	250	409.5	4
		9	6	3,6	3,7	42.9	273	409.5	4
		10	5	3,5	3,7	42.9	270	470.0	4
		13	4	2,4	2,6	33.3	272	533.0	4
		14	4	2,4	2,6	33.3	308	588.0	4
MANSOOR	11	48	4	2,4	3,5.5	54.5	3456	7776.0	4
		62	3	2,3	2,4	50.0	4712	9362.0	4
		94	2	1,2	1,3	33.3	4982	10152.0	4
MITCHELL	21	14	8	6,8	6,10	60.0	854	1526.0	4
		15	8	5,7	6,10	60.0	800	1680.0	4
		21	5	4,5	4,8	50.0	1090	1890.0	4
		26	5	3,5	3,6.5	46.2	1274	2223.0	4
		35	3	3,3	3,4	75.0	1976	2957.5	3
HESKIA	28	138	8	4,8	4,10.5	38.1	43056	104397.0	4
		205	5	3,5	3,8	37.5	65906	139810.5	4
		216	5	3,5	2,8	25.0	73008	129924.0	4
		256	4	2,4	2,7	28.6	67840	153344.0	4
		324	4	2,4	2,6	33.3	107892	193104.0	4
SAWYER30	30	25	14	4,13	7,18	38.9	2973	7225.0	6
		27	13	4,12	6.5,15	43.3	3076	7431.8	5
		30	12	4,11	6,14	42.9	3330	7950.0	5
		33	11	3,10	6,14	42.9	3051	8761.5	5
		36	10	3,9	6,13	46.2	3240	9486.0	5
KILBRID	45	56	10	3,10	3.5,14.5	24.1	7000	19852.0	6
		57	10	3,10	3,14.5	20.7	7209	20491.5	6
		62	9	3,9	3,13	23.1	7998	21328.0	5
		69	8	2,8	2.5,11.5	21.7	6899	21786.8	5
		79	7	2,7	2.5,11	22.7	8294	25023.3	5
TONGE70	70	160	23	8,22	8,29.5	27.1	154400	403520.0	5
		168	22	8,21	8,30	26.7	163296	422016.0	5
		176	21	7,20	8,28	28.6	156816	439032.0	5
		185	20	7,19	8,27.5	29.1	166404	455192.5	5
		195	19	7,18	7,24.5	28.6	177934	439238.0	5
ARC83	83	3786	21	11,20	12,25	48.0	115654728	205104500.0	4
		3985	20	11,19	12,23	52.2	123792027	210840000.0	4
		4206	19	10,18	10,21.5	46.5	124493394	206861500.0	4
		4454	18	10,17	10,20.5	48.8	134782494	222246000.0	4
		4732	17	9,16	8,20	40.0	135841524	216399000.0	4
ARC111	111	5755	27	11,27	12,39.5	30.4	192576682	599806000.0	5
		5785	27	11,26	12,39	30.8	193597912	600683000.0	5
		6016	26	11,25	12.5,38	32.9	208219776	649234500.0	5
		6267	25	10,24	12,36.5	32.9	205137706	659191000.0	5
		6540	24	10,23	12,37.5	32.0	222209580	710823000.0	5
total					293.5,759	45.95			

As seen in this table, GA has reached the optimum solution for all instances of MERTENS, BOWMAN and JAESCHKE (the entire instance solved to optimality using lingo 11). Another observation is the significant decrease in the number of stations in comparison to *SALBP*-1, this result in better space utilization. Space utilization can be defined as the proportion of one station used by one worker and can be calculated by the following equation [16, 18]:

$$f = \frac{m}{tw} \qquad (7)$$

In this equation f is the space utilization factor, m is the number of stations and tw is the number of workers. This factor ranges between 1 and $1/tw$ and is of special importance if there are space constraints in the production floor which may happen because of the building design or redesigning the line to produce a new product.
The multi-manned system results in a shorter physical line length and improves the space utilization. Because in this system the same number of workers can be allocate to fewer stations comparing to the traditional approach. In Table 3, in many instances the space utilization factor has improved and for all examples the average space utilization factor is 45.95 percent. This means that the required space has reduced to 45.95 percent of its previous value for the traditional approach.

6. Conclusions and Future Research

MALs are a new type of lines in which there can be more than one worker in each station working on the same work-piece. This type of line is very common in manufacturing of large-sized products e.g. vehicle final assembly. MALs have several advantages over the traditional lines which include reducing the length of the line and better utilization of the tools and machinery in stations. On the other hand this type of lines results in reducing the work in process and throughput time which is of high priority for production managers.

In the classical *MALBP* the objective is to minimize the manpower needed to manufacture one product unit. Apart from the manpower, other cost drivers like wage rates or machinery are neglected in this classical view of the problem. But due to the high competition in the current production environment, reducing the production costs and increasing utilization of available resources are very important issues for manufacturing managers.

Although minimizing the costs of production is of major importance in practice, there has not been sufficient consideration in the literature of MAL. In this paper the MALBP was considered with the aim of minimizing the total costs of one production unit. For this aim a mathematical formulation was presented. Furthermore, in order to be able to solve the medium- and large-size scales of the problem, several heuristics and a GA were proposed. Several examples were solved to show the effectiveness of the proposed model and proposed algorithms.

However, since the tasks are performed by human being, it is reasonable to assume task times be stochastic. Therefore the current research can be extended to the stochastic environments in MALs and incompletion costs can be additionally considered. Also developing other heuristic or meta-heuristics such as Tabu search or ant colony optimization to solve the introduced model are recommended for future research in this area.

7. References

[1] Agpak, K.K. (2005). Assembly line balancing: two resource constraint cases. International Journal of Production Economics, 96, 129–140.

[2] Akpinar, S., Bayhan, G.M. (2011). A hybrid genetic algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints. Engineering Applications of Artificial Intelligence, 24, 449−457.

[3] Amen, M. (1997). Ein exaktes Verfahren zur kostenorientierten Fliebbandabstimmung, in: U. Zimmermann et al.., (Eds.), Operations Research Proceedings 1996, Springer, Berlin, pp. 224-229.

[4] Amen, M. (2000a). An exact method for cost-oriented assembly line balancing. International Journal of Production Economics 64, 187–195.

[5] Amen, M. (2000b). Heuristic methods for cost-oriented assembly line balancing: A survey. International Journal of Production Economics, 68, 1–14.

[6] Amen, M. (2001). Heuristic methods for cost-oriented assembly line balancing: A comparison on solution quality and computing time. International Journal of Production Economics, 69, 255–264.

[7] Amen, M. (2006). Cost-oriented assembly line balancing: Model formulations, solution difficulty, upper and lower bounds. European Journal of Operational Research, 168, 2006, 747–770.

[8] Andrés, C., Miralles, C., Pastor, R. (2008). Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. European Journal of Operational Research, 187 (3), 1212–1223.

[9] Bahalke, U., Yolmeh, A.M., Shahanaghi, K. (2010). Meta-heuristics to solve single-machine scheduling problem with sequence-dependent setup time and deteriorating jobs. Int J Adv Manuf Technol, 50, 749–759. DOI: 10.1007/s00170-010-2526-5.

[10] Bartholdi, J.J. (1993). Balancing two-sided assembly lines: A case study. International Journal of Production Research, 31, 2447–2461.

[11] Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. Management Science , 32, 909–932.

[12] Boysen, N., Fliedner M., Scholl A. (2007). A classification of assembly line balancing problems. Eur J Oper Res 183:674–693.

[13] Buxey, G.M., (1974). Assembly line balancing with multiple Stations. Management Science, 20, 1010–1021.

[14] Cevikcan, E., Durmusoglu, BM., Unal, ME. (2009). A team-oriented design methodology for mixed model assembly systems. Comput Ind Eng, 56,576–599.

[15] Chang, H. J., Chang, T. M. (2010). Simultaneous Perspective-Based Mixed-Model Assembly Line Balancing Problem. Tamkang Journal of Science and Engineering, 13 (3), 327_336.
[16] Corominas, A., Ferrer, L., Pastor, R. (2010). Assembly line balancing: general resource-constrained case, International Journal of Production Research, 49(12), 3527 – 3542,
DOI: 10.1080/00207543.2010.481294.
[17] Dimitriadis, SG. (2006). Assembly line balancing and group working: a heuristic procedure for workers' groups operating on the same product and workstation. Comput Oper Res 33:2757–2774
[18] Erel, E., Sarin, S. (1998). A survey of the assembly line balancing procedures. Production Planning and Control, 9, 414 – 434.
[19] Fattahi, P., Roshani, A., Roshani, A. (2011). A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. Int J Adv Manuf Technol, 53, 363–378.
[20] Gehrlein, W. (1986). On methods for generating random partial orders.Operation s Research Letters 5 (6), 285–291.
[21] Grangeon, N., Leclaire, P., Norre, S. (2011). Heuristics for the re-balancing of a vehicle assembly line. International Journal of Production Research, DOI: 10.1080/00207543.2010.539025 (to appear)
[22] Hahn, R. (1972). Produktionsplanung bei Linienfertigung, Walter de Gruyter, Berlin.
[23] Helgeson, W.B., Birnie, D.P. (1961). Assembly line balancing using the ranked positional weight technique. Journal of Industrial Engineering, 12, 394-398.
[24] Malakooti, B. (1991). A multiple criteria decision making approach for the assembly line balancing problem. International Journal of Production Research, 29, 1979–2001.
[25] Malakooti, B. (1994). Assembly line balancing with buffers by multiple criteria optimization. International Journal of Production Research, 32, 2159–2178.
[26] Malakooti, B., Kumar, A. (1996). An expert system for solving multi-objective assembly line balancing problems. International Journal of Production Research, 34, 2533–2552.
[27] Martino, L., Pastor, R. (2010). Heuristic procedures for solving the general assembly line balancing problem with setups. International Journal of Production Research, 48 (6), 1787 – 1804. DOI: 10.1080/00207540802577979
[28] Miltenburg, J., Wijngaard, J. (1994). The U-line line balancing problem. Management Science, 40, 1378–1388.
[29] Pastor, R. (2011). LB-ALBP: the lexicographic bottleneck assembly line balancing problem. International Journal of Production Research, 49 (8), 2425 — 2442, DOI: 10.1080/00207541003705856
[30] Pinto, P.A., Dannenbring, D.G., Khumawala, B.M. (1983). Assembly line balancing with processing alternatives: an application. Management Science, 29, 817–830.
[31] Rekiek, B., Dolgui, A., Delchambre, A., Bratcu, A. (2002). State of art of optimization methods for assembly line design. Annual Reviews in Control, 26, 163–174.
[32] Rosenberg, O., Ziegler, H. (1992). A comparison of heuristic algorithms for cost-oriented assembly line balancing, Zeitschrift fur Operations Research, 36, 477-495.
[33] Ruiz, R., Maroto, C., Alcaraz, J. (2005). Solving the flowshop scheduling problem with sequence-dependent setup times using advanced metaheuristics. Eur J Oper Res, 165, 34–54.
[34] Ruiz, R., Maroto, C., Alcaraz, J. (2006). Two new robust genetic algorithms for the flowshop scheduling problem. Omega, 34, 461–476.
[35] Scholl, A. (1999). Balancing and sequencing assembly lines, 2nd edn. Physica, Heidelberg.
[36] Scholl, A., Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. European Journal of Operational Research, 168, 666-693.
[37] Seyed-Alagheband, S. A., Fatemi Ghomi, S.M.T., Zandieh, M. (2011). A simulated annealing algorithm for balancing the assembly line type 2 problem with sequence-dependent setup times between tasks.
[38] Sivanandam, SN., Deepa, SN. (2008). Introduction to genetic algorithms. Springer, New York.
[39] teffen, R. (1977). Produktionsplanung bei Fliebbandfertigung, Gabler, Wiesbaden.
[40] Tonge, F.M. (1965). Assembly line balancing using probabilistic combinations of heuristics. Management Science, 11, 727-735.
[41] Vallada, E., Ruiz, R. (2010). Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem. Omega, 38, 57–67. doi:10.1016/j.omega.2009.04.002.
[42] Yolmeh, A., Kianfar, F. (2012). An efficient hybrid genetic algorithm to solve assembly line balancing problem with sequence-dependent setup times. Computers & Industrial Engineering, 62, 936−945.